



National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file *Votre référence*

Our file *Notre référence*

NOTICE

The quality of this microform is heavily dependent upon the quality of the original thesis submitted for microfilming. Every effort has been made to ensure the highest quality of reproduction possible.

If pages are missing, contact the university which granted the degree.

Some pages may have indistinct print especially if the original pages were typed with a poor typewriter ribbon or if the university sent us an inferior photocopy.

Reproduction in full or in part of this microform is governed by the Canadian Copyright Act, R.S.C. 1970, c. C-30, and subsequent amendments.

AVIS

La qualité de cette microforme dépend grandement de la qualité de la thèse soumise au microfilmage. Nous avons tout fait pour assurer une qualité supérieure de reproduction.

S'il manque des pages, veuillez communiquer avec l'université qui a conféré le grade.

La qualité d'impression de certaines pages peut laisser à désirer, surtout si les pages originales ont été dactylographiées à l'aide d'un ruban usé ou si l'université nous a fait parvenir une photocopie de qualité inférieure.

La reproduction, même partielle, de cette microforme est soumise à la Loi canadienne sur le droit d'auteur, SRC 1970, c. C-30, et ses amendements subséquents.

University of Alberta

The High-Performance Distributed Queue Protocol for High Speed MANs.

by



Ashvaneer Bissonauth

A thesis
submitted to the Faculty of Graduate Studies and Research
in partial fulfillment of the requirements for the degree
of Master of Science

Department of Computing Science

Edmonton, Alberta

Spring 1993



National Library
of Canada

Acquisitions and
Bibliographic Services Branch

395 Wellington Street
Ottawa, Ontario
K1A 0N4

Bibliothèque nationale
du Canada

Direction des acquisitions et
des services bibliographiques

395, rue Wellington
Ottawa (Ontario)
K1A 0N4

Your file *Votre référence*

Our file *Notre référence*

The author has granted an irrevocable non-exclusive licence allowing the National Library of Canada to reproduce, loan, distribute or sell copies of his/her thesis by any means and in any form or format, making this thesis available to interested persons.

L'auteur a accordé une licence irrévocable et non exclusive permettant à la Bibliothèque nationale du Canada de reproduire, prêter, distribuer ou vendre des copies de sa thèse de quelque manière et sous quelque forme que ce soit pour mettre des exemplaires de cette thèse à la disposition des personnes intéressées.

The author retains ownership of the copyright in his/her thesis. Neither the thesis nor substantial extracts from it may be printed or otherwise reproduced without his/her permission.

L'auteur conserve la propriété du droit d'auteur qui protège sa thèse. Ni la thèse ni des extraits substantiels de celle-ci ne doivent être imprimés ou autrement reproduits sans son autorisation.

ISBN 0-315-82216-3

UNIVERSITY OF ALBERTA

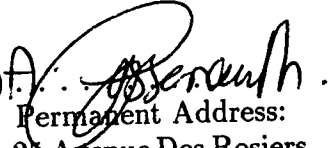
RELEASE FORM

NAME OF AUTHOR: Ashvance Bissonauth
TITLE OF THESIS: The High-Performance Distributed Queue Protocol for
High Speed MANs.

DEGREE: Master of Science
YEAR THIS DEGREE GRANTED: 1993

Permission is hereby granted to the UNIVERSITY OF ALBERTA LIBRARY to reproduce single copies of this thesis and to lend or sell such copies for private, scholarly or scientific research purposes only.

The author reserves other publication rights, and neither the thesis nor extension extracts from it may be printed or otherwise reproduce without the author's written permission.

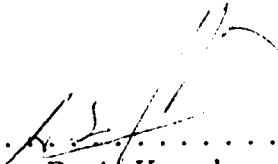
(Signed) 
Permanent Address:
25 Avenue Des Rosiers,
Quatre-Bornes.
MAURITIUS.

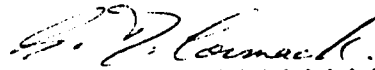
Date: 19/03/93

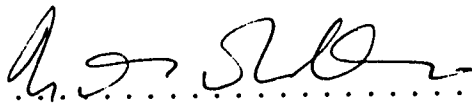
UNIVERSITY OF ALBERTA

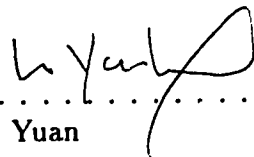
FACULTY OF GRADUATE STUDIES AND RESEARCH

The undersigned certify that they have read, and recommend to the Faculty of Graduate Studies and Research, for acceptance, a thesis entitled **The High Performance Distributed Queue protocol for High-Speed Metropolitan Area Networks** submitted by Ashvaneer Bissonauth in partial fulfillment of the requirements for the degree of Master of Science.


.....
Supervisor: Dr A. Kamal


.....
External: Dr G.D. Cormack (Dept. of Elect. Eng.)


.....
Examiner: Prof. U.M. Maydell


.....
Chair: Dr L.Y. Yuan

Date: *March 2/93*

Dedication

In recognition of the unconditional love and belief that my family have shown me over the past six years, I dedicate this thesis to them.

To my father, Oomahshankar.

whose dream it was that I should achieve an
M.Sc. degree.

To my mother, Kamlaotee,
and grandmother, Chando,

whose emotional support comforted me.

And finally to my sisters,
Asvina and Aruna,

who have missed me, written to me
and included me in their lives even though
I was away.

I thank you all.

ABSTRACT

The IEEE 802.6 committee has adopted the DQDB as the standard metropolitan area network. Studies of this network have shown that the behavior of the DQDB medium access control is unpredictable in the sense that bandwidth may be distributed unfairly among nodes. This anomalous behavior is not only present under unipriority type of operation, but also between nodes operating at different priority levels. Remedial corrections to the MAC protocol have been proposed in the literature, in addition to alternative network protocols.

This thesis proposes a novel network protocol, the High-Performance Distributed Queue (HPDQ). The network is based on the double bus topology and is capable of operating at a rate of multi gigabits per second. The objectives behind the design of this protocol are to provide equitable access and reliable operation. Fluctuations in network activities should have negligible impact on the network operation by redistributing the bandwidth among the active nodes within the same priority level. The protocol also implements priority by allowing higher priority nodes to discriminate against lower priority ones in terms of bandwidth offering. The protocol should also subscribe to the class of network protocols offering integrated access and service of multiple services.

The principle of operation of HPDQ protocol is fundamentally differ-

ent from other protocols. It requires nodes to maintain essential information about activities downstream, such as the number of active nodes, and their aggregate bandwidth requirements. Additionally, the network nodes must be equipped with active taps, which allow the modification of data observed on the bus. Through the proper handling of this pair of information items by the protocol, the protocol is capable of satisfying the above objectives.

To demonstrate the validity of the above claims, a simulation study was undertaken to assess both the steady state performance and the transient behavior of the HPDQ. By analysing several important scenarios, it is shown that the results establish that the objectives of the protocol have been met.

Acknowledgments

I would like to express my gratitude to my supervisor Dr A. E. Kamal for his exemplary guidance in my research work, for devoting quality time into proof reading my thesis and for providing me with financial support. My thanks to the members of my examining committee for their comments and suggestions to improve the quality of my thesis.

The help that I have received from the staff of the Department of Computing Science should not be left unmentioned. I would like to thank the Department for their support without which the completion of this thesis might not have been possible.

My sincere thanks also go to my father who gave me the opportunity to study in Canada. His dedication in writing to me weekly for the past six years, his understanding and guidance have provided me with great moral and emotional support at times when I most needed them.

Lastly, I would like to thank my friends Aditya K. Ghose, Suryanil Ghosh, Ramesh S. Sankaranayana, Florence Lehmann and Audrey Smith for making life outside the university so pleasant. Their friendship is most appreciated. They will always have a very special place in my heart.

Contents

1	INTRODUCTION	1
1.1	Computer Networks	4
1.2	Introduction to MANs	5
1.2.1	Architecture of a MAN	5
1.2.2	Early proposals for a MAN standard	7
1.2.3	Media Access Protocol for DQDB	9
1.3	Problems with the DQDB MAC protocol	15
1.3.1	Heavy loading by an initial heavy user	15
1.3.2	Heavy Loading Initiated by a group of users	17
1.3.3	Node Stopping Transmission	18
1.3.4	Unfair behavior in the priority mechanism	18
1.3.5	Bandwidth Balancing Mechanism	21
1.4	Other Proposals	23
1.4.1	Distributed Queue Multiple Access (DQMA)	24

1.4.2	Cyclic Reservation Multiple Access protocol	26
1.4.3	Cyclic Reservation Multiple Access II protocol	28
1.4.4	Fair Distributed Queue protocol	30
1.5	Summary	32
1.6	Thesis Overview	33
1.7	Thesis Outline	34
2	MEDIUM ACCESS PROTOCOL FOR HPDQ	36
2.1	Objectives	36
2.2	Description of HPDQ	37
2.3	The Enhanced HPDQ Protocol	44
2.4	Network Aspects	50
2.4.1	Performance issues	50
2.4.2	Implementation Cost and Complexity	52
2.5	Summary	54
3	PERFORMANCE ASPECTS	55
3.1	Simulation of the Basic Protocol	57
3.1.1	Steady state analysis	57
3.1.2	Transient analysis	60
3.2	Simulation of the Enhanced Protocol	66
3.2.1	Transient analysis	66
3.3	SUMMARY	67

4	Priority Implementation	69
4.1	Introduction and Background	69
4.2	Priority Scheme for IIPDQ	73
4.3	The Buffering System	80
4.4	Summary	86
5	Performance of the HPDQ priority scheme	88
5.1	Nodes with different starting time and positions	89
5.2	Nodes of the same priority level	90
5.3	Effect of distances on performance of nodes under heavy load .	97
5.4	SUMMARY	100
6	CONCLUSIONS and DIRECTIONS for FUTURE RESEARCH	103

List of Figures

1.1	MAN with access and transport subnetworks	6
1.2	Networks interconnection	10
1.3	DQDB network configuration	11
1.4	DQDB Access Control Field Format	13
1.5	Transfer of segments	13
1.6	Nodes of different priorities	20
2.1	HPDQ network configuration.	38
2.2	Finite State Machine for Basic HPDQ protocol.	39
2.3	Slot format for basic HPDQ.	40
2.4	Basic operation of HPDQ	42
2.5	Node 2 stops updating at completion of its transmission . . .	43
2.6	Bandwidth loss due to stopping of transmission.	45
2.7	Slot format for enhanced HPDQ.	46
2.8	Finite State Machine of Enhanced HPDQ protocol.	47
2.9	Operation of the CumReq_ctr counters.	48

2.10	Bandwidth distribution due to node stopping transmission. . .	49
3.1	Transient utilization of nodes starting from left to right	62
3.2	Transient behavior of nodes at different distances from each other.	63
3.3	Transient utilization of node 5 leaving the system.	65
3.4	Transient utilization of node 5 leaving the system (Enhanced HPDQ).	67
4.1	Finite state machine for nodes' operation on the reverse bus. .	75
4.2	Finite state machine for nodes' operation on the forward bus. .	76
4.3	HPDQ Access Control Field format.	77
4.4	High priority node leaving the system	85

List of Tables

3.1	Maximum bus length for throughput fairness	58
3.2	Node utilization.	58
3.3	Access Delay	59
3.4	Utilization for different initial heavy users.	60
3.5	Summary of system.	62
3.6	Utilization at 660	62
3.7	Utilization at 680	62
3.8	Summary of system.	63
4.1	Summary of α	79
4.2	Summary of AND_ctrs update	79
5.1	Average utilization and average access delay for various speeds.	90
5.2	Utilization for scenario 1.	92
5.3	Average access delay for HPDQ under scenario 1.	93

5.4	DQDB with priority and without Bandwidth Balancing Mechanism.	93
5.5	DQDB protocol with Global priority scheme.	94
5.6	Utilization for HPDQ under scenario 2.	95
5.7	Transfer time for HPDQ under scenario 2.	96
5.8	Internode distances of network of 5 nodes.	98
5.9	Performance results for nodes at different internode distances.	98
5.10	DQDB: Throughput distribution for different priority configurations (seven users, balancing ratio 8/9).	100
5.11	HPDQ: Throughput distribution for different priority configurations.	101

Chapter 1

INTRODUCTION

Since the beginning of time, man has been pushing back the barriers of reality. We have migrated from the stone age, through the machine age into the information age. Indeed, the twentieth century is about gathering, processing and distributing information. The unprecedented growth of the computer industry is a definite manifestation of a new era.

During the first couple of decades of this computer age, computers were of a centralized nature. They were accessible to a small privileged portion of the world population. Today, besides becoming more accessible, computers are getting more decentralized. The decentralization is closely linked to the merging of computers and communications. This bonding has permanently influenced the way the computer system is organized and has led to the merger between the communication and the computer industries

creating what has become known as the computer network.

In general terms a computer network can be defined as an interconnection of a collection of autonomous computers [18] and its purpose is to share resources, provide a higher reliability and extend the scale of cost effectiveness.

In the early days of computers, the computer industry had to advertise for the services that computers are able to provide as people were unaware of its potential. Today, the computer industry has overwhelmed people who are willing to pay for the many services that it provides [1]. These services can be integrated together and an International Committee has been set up to look into this integration. The CCITT (Consultive Committee for International Telephony and Telegraph) has been developing and standardizing the ISDN (Integrated Services Digital Network) since mid 1970's [5, 21, 1] which has eventually evolved into BISDN (Broadband Integrated Services Digital Network) [21, 6, 1]. The following (incomplete) list depicts the services that BISDN would provide to the different applications in the future:

Communication of data, text, graphics.

- Data transfer (burst, stream)
- Document transfer
- document filing and retrieval

Person to person video communications.

- Video telephony (including showing documents etc)

- Video conference
- Broadband message handling

Access to video information.

- Broadband videotex
- Video on demand

Broadcast of programs and data.

- Common TV
- Pay TV (pay per channel, pay per view)
- High-definition television (HDTV)
- Cabletext

In order to be able to provide this impressive list of services in a cost effective manner, a unified data transport service must support all these applications. Among other alternatives, a new multiplexing scheme known as Asynchronous Transfer Mode(ATM) [16, 3, 20], has been adopted as the transport mechanism BISDN. ATM can be thought of as a multiplexing and switching techniques confined to layer 1 and basic functions of layer 2 of the OSI model. ATM is based on the use of fixed length packets, 53 octets called cells. Out of these, 48 octets are used for data, while 5 octets are used as a header. One of the main features of ATM is its use of highly simplified protocols, which is useful in high speed network.

1.1 Computer Networks

From a geographical-coverage viewpoint, computer networks can be classified into three main classes, namely Local Area Networks(LAN), Wide Area Networks(WAN) and the more recent Metropolitan Area Networks(MAN). A LAN is a communication network that provides interconnection of a variety of devices such as computers, terminals and peripherals within a limited geographical area(e.g. LAN can be used to connect several machines on different floors in a building). LANs are based on a shared medium that runs at a sufficiently high speed to accommodate needs of many users at once. LANs differ from the other networks in that they span shorter distances, typically a few kilometers, have high data rates, low error rates, simpler routing, are owned and used by a single organization and have lower communication costs [4]. At the other end of the spectrum, we have the WAN which spans several hundred kilometers. For WANs the RS-232 interface has served as a fixed point through which the great bulk of data transmission has passed in the past two decades. WANs have lower data rates, more complex routing mechanisms and are not necessarily used by the central body that owns it. LANs and WANs shall not be discussed any further as the rest of this thesis is concerned with MANs.

1.2 Introduction to MANs

Metropolitan Area Networks inherit characteristics from LANs and WANs. MAN started as an extension of LAN, but its size and scope dictate that it be operated by a central body. In essence, a MAN is a large LAN with access protocols less sensitive to network size than those used by LANs. As MAN will serve more users, interconnect both LANs and large computers, it must provide higher bandwidth. With high data rates, it will be capable of supporting data, voice and video communication services.

In this section the MAN architecture shall be described followed by a review of some of the earlier proposals for a MAN standard. Thereafter, the current standard for metropolitan area networks shall be discussed and end with more recent proposals for medium access protocols for MANs.

1.2.1 Architecture of a MAN

MAN installation can be either public or private [14]. The private MAN is analogous to the dedicated line; it is leased to a single company to connect a number of sites in a metropolitan area. Large organizations can afford a dedicated line but this is not justifiable for small ones. A public MAN is divided into two parts: an access network and a transport network, as shown in Figure 1.1., [14]. The access network terminates at the customers' premises

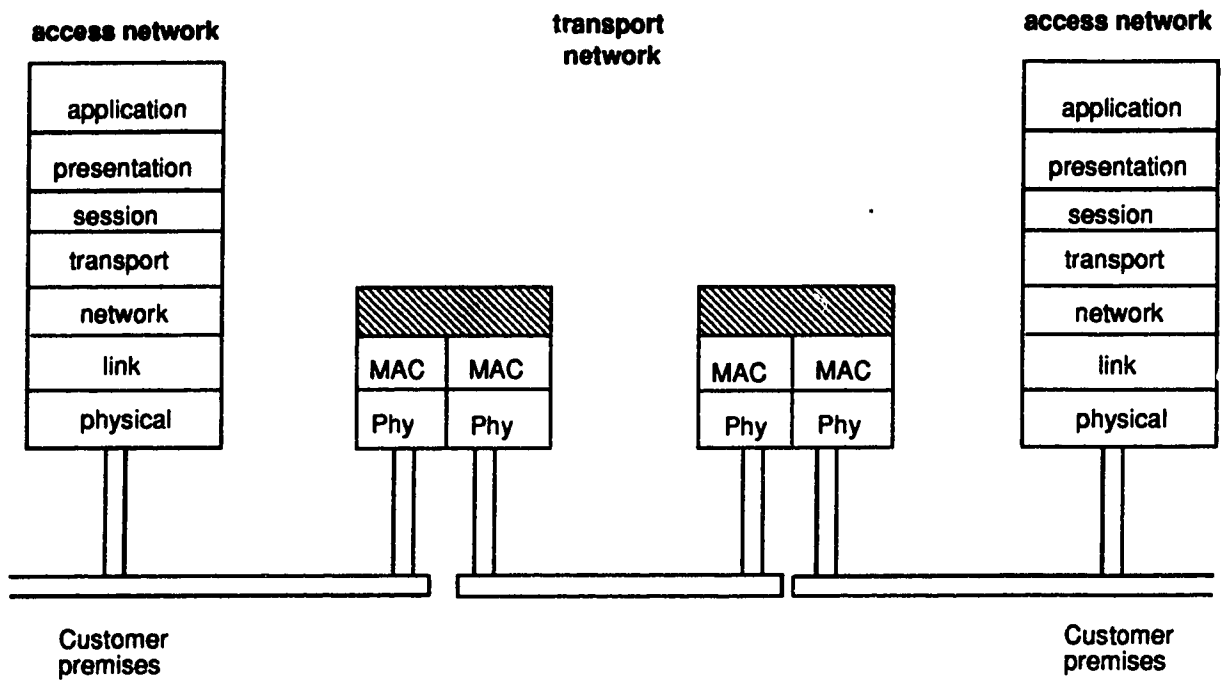


Figure 1.1: MAN with access and transport subnetworks

and is bridged to the transport network at the Medium Access Control level. The transport network is therefore invisible to the customer's equipment. The bridge must recognize present addresses on the transport network and selectively pass the data on to the access network. Similarly, outgoing data are transferred to the transport network. The access network can be another segment of the MAN network, or can be an IEEE 802 LAN (such as covered by standard IEEE 802.3 which applies to the ETHERNET system) that internally serves the building's occupants. Because of the growing importance of MANs, the need for a standard for metropolitan area networks was recognized as early as 1981 by the IEEE's Project 802, the standardization body for LANs. The specific standard for MANs is now designated IEEE 802.6. Several proposals to be covered have been considered for inclusion in a MAN standard over the last decade. Some of those proposals are reviewed next.

1.2.2 Early proposals for a MAN standard

Unless a standard interface between the network and the customer premises equipment exists, interconnection of different vendor equipment becomes impossible. MANs are therefore being standardized under Project 802.6 whose goal is to come up with a MAN standard that will provide a two-way interchange of digital bit streams using a shared medium between

nodes located within an area that is typically up to 50 kilometers in diameter. A standard should support services that require guaranteed bandwidth and constrained delay. The range of transmission rates should extend from 1 Mbps to the appropriate limits of the medium used.

The first proposals for MANs were based on time division multiplexing protocols, similar to those used by multiple ground stations to access a satellite channel. Unfortunately, this protocol's original growth or profitability expectation were never reached. Another proposal that was also based on a radio system which is already in production was brought in by Motorola but was rejected because its data rate fell below the Project 802 threshold of 1 Mbps. Yet, another proposal used the FDDI token ring protocol for data packets and superimposed a set of 64 kilobits/sec slots for isochronous traffic. The fixed 100 Mbps speed of FDDI was modified to match the speeds of the telephone industry's transmission hierarchy, ranging from 1.544 Mbps for T1 to approximately 155 Mbps which is the T channel of the SONET standard.

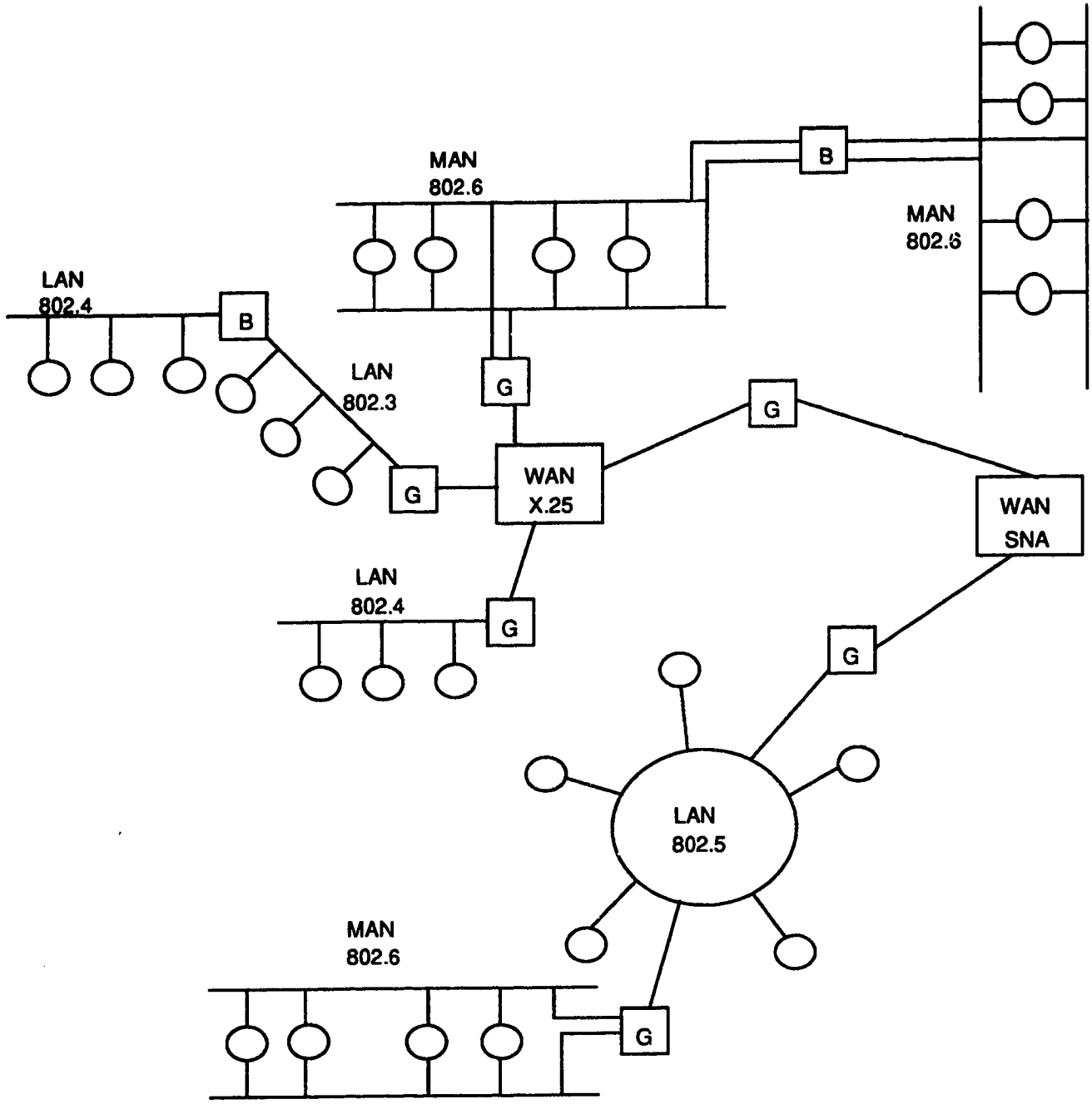
Telecom Australia submitted a competing proposal at the same time. The protocol that they put forward, called QPSX for Queued Packet for Synchronous Exchange, was unique. It is based on two unidirectional busses running in opposite directions. Writing to the bus is only possible in unused slots. Scheduling is done by sending reservations *upstream* when a station wants to transmit *downstream*. Each station has a pair of counters which forces stations to transmit under specific conditions only, as will be discussed

in the next section 1.2.3.

In November 1987, a consensus emerged within the IEEE Project 802 to use a modified version of QPSX, better known as the Distributed Queue Dual Bus(DQDB) as the medium access protocol to be used for standardizing MANs. A few years later, the IEEE 802.6 committee approved DQDB network as the standard Metropolitan Area Network. The media access protocol for DQDB will now be described in detail, followed by a discussion of some of its anomalous aspects of behavior. Figure 1.2 shows how the DQDB network fits in the communication system. In this example, it is shown that two 802.6 MANs can be connected by bridges. Gateways are used to connect 802.6 MAN to LAN or WAN.

1.2.3 Media Access Protocol for DQDB

The network configuration of the DQDB protocol is shown in Figure 1.2. It consists of a pair of unidirectional busses serving opposite directions. Since the operation of the node is the same on both busses, only the operation of the node on one bus, bus A, shall be considered without loss of generality. Bus A shall be referred to as the forward bus, while bus B shall be called the reverse bus. The operation is slotted and at the head of each bus a special station referred to as the head end station is responsible for



G GATEWAYS
B BRIDGES

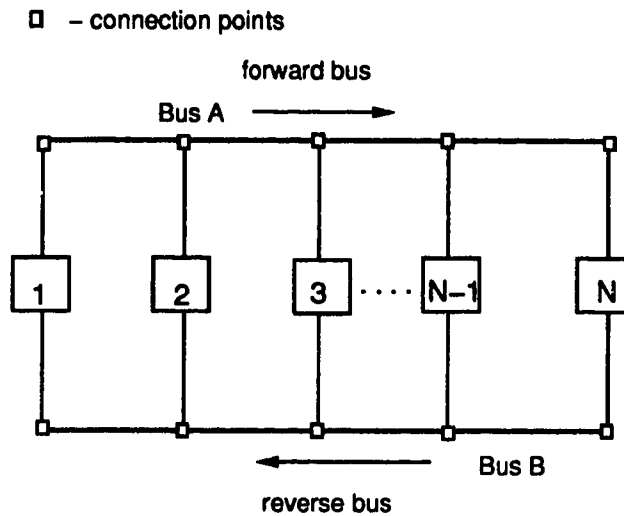


Figure 1.3: DQDB network configuration

generating frames which are subdivided into fixed length slots. The duration of a frame is $125 \mu s$ (sampling period used in digital telephony). A frame is subdivided into fixed length slots. The slot format is shown in Figure 1.4. The busy bit indicates whether a slot is used or unused. The SL_Type bit indicates whether the slot is a Queued Arbitrated (used for asynchronous transmission) or Pre_Arbitrated (used for isochronous transmission). The PSR bit indicates whether the segment in the previous slot may be cleared or not. A request bit is set to indicate that a station is requesting a slot on the other bus. Three such bits exist to implement three different priority levels. The slots can be used to transmit both synchronous and asynchronous traffic. The network controller (head station) reserves some slots of a frame

for synchronous use and allocates them to stations that have requested synchronous packets. Slots not reserved for synchronous circuits are available for packet switched communication. The access to the slots is determined according to the DQDB medium access protocol. In this section only Queued Arbitrated traffic is considered.

The unipriority DQDB medium access protocol is discussed first. On both busses, a node can send data to downstream nodes and receive data from upstream nodes, or vice versa. Therefore, nodes need to maintain information about their relative positions on the network. Existence of the two busses running in opposite directions allows nodes to send requests on one bus to inform the upstream stations on the other bus of their intention to send data on the latter. User data is segmented to fit into slots and are kept in a local queue. Segments are transmitted on a FIFO basis. The segment at the top of the local queue is moved to the transmission queue, once the latter is empty. At the same time a request is enqueued for transmission on the reverse bus. The transmission queue can hold only one segment, Figure 1.5.

In the DQDB protocol [3] a distributed queue is implemented by a pair of Request and Countdown counters for each bus. When a node has a segment ready for transmission, it enqueues a self request for transmission on the reverse bus. The request is transmitted in the next empty request field on the reverse bus. A node also observes request bits on the reverse bus and

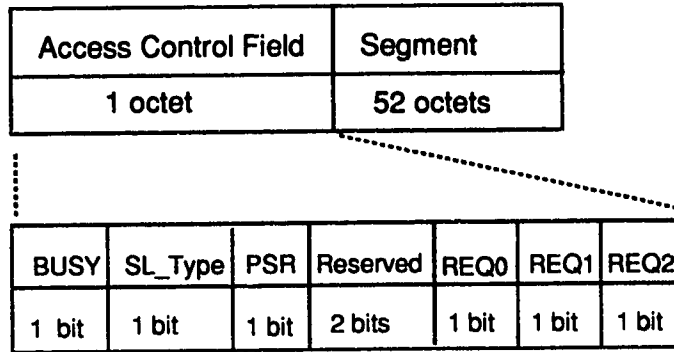


Figure 1.4: DQDB Access Control Field Format

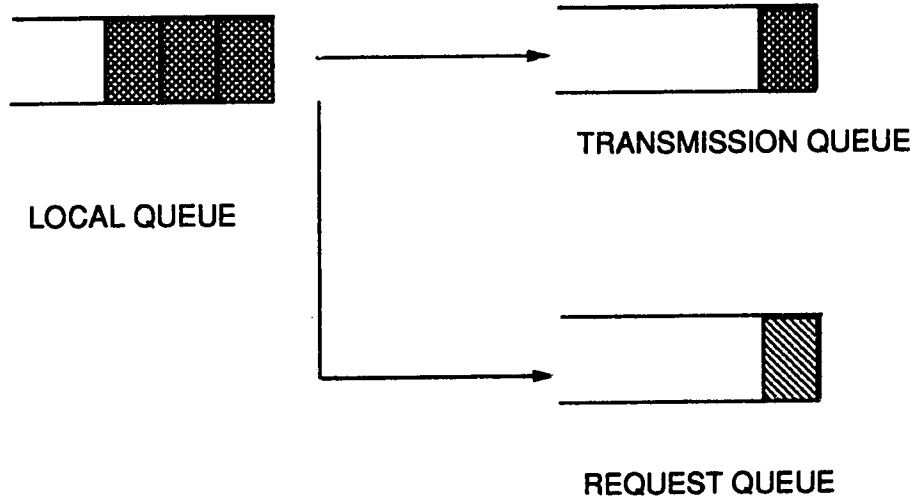


Figure 1.5: Transfer of segments

increments the value of its Request Counter by 1 to keep track of the number of segments ready for transmission by downstream stations. This implements a distributed queue. To insert its own segment in the queue, a node transfers the content of its Request Counter to the Countdown counter, and resets the former to zero. An empty slot on the forward bus causes the Countdown Counter to be decremented by one until it reaches zero. At this point, the node is allowed to transmit its segment in the next available empty slot. After doing so, another segment from the local queue may be moved to the transmission queue and the whole process will be repeated. Under light load, the basic DQDB protocol behaves reasonably well and is fair to all nodes. Unfortunately problems emerge as the load becomes heavier [22, 27, 8].

According to the latest version of the standard, DQDB supports three priority levels [3]. This is achieved by placing three request bits in each slot and having a set of counters for each level of priority. The basic DQDB protocol is extended by letting a free slot decrease the counters of all three priority levels. At the same time, a passing request of priority i increments the request counter for that priority level, but increments all Request Counters of lower priorities for which no data is scheduled and all Countdown Counters for which data is scheduled. Just like in the basic protocol, under light load the priority mechanism works but is inefficient under heavy load [22, 27, 8, 12]. Thus, the priority mechanism fails to work properly. In the next section we shall elucidate on these problems and discuss some of the

proposed solutions.

1.3 Problems with the DQDB MAC protocol

Under light load, the DQDB protocol behaves reasonably. Unfortunately, its behavior is not flattering at heavier loads. We shall define a network to be under heavy load if the number of segments queued up for transmission is enough to keep the entire bus busy. In the following section, we have chosen various scenarios to bring up the drawbacks of the DQDB protocol.

1.3.1 Heavy loading by an initial heavy user

In this section, heavy load nodes at different positions on the network with different starting times are discussed. These cases were investigated in references [22, 10, 27, 2, 7, 13]. As reported in those references, the initial heavy user is definitely the largest consumer of the bandwidth. This unfair behavior becomes more severe the closer the initial heavy user is to the head-end. Besides, the faster the speed or the shorter the bus length, the higher the throughput of the initial heavy user node.

The reason for this unequal distribution of throughput is due to the

incremental effect which is explained in detail in [22, 27, 13]. Basically, after the initial node has flooded the forward bus with its segments and the reverse bus with its requests, the other nodes are forced to transmit in a certain pattern. Upstream stations from the initial heavy user record requests while transmitting their segments. When upstream station let empty slots pass by on the forward bus for downstream stations, upstream stations record more requests on the reverse bus from initial heavy user stations. Consequently, upstream stations transmissions are scheduled further apart after each of their transmissions. This process is repeated for some time and is subsequently partially suppressed by requests sent from nodes downstream from the initial heavy user. Eventually these two opposing forces stabilize the system and provide the initial heavy user with the largest share of throughput. The steady state throughput rate for two nodes can be described by the following equations [10]:

$$\gamma_1 = 2/(2 - D - C(\delta) + ((D - C(\delta) + 2)^2 + 4DC(\delta))^{1/2})$$

$$\gamma_2 = 1 - \gamma_1$$

γ_1 and γ_2 are the nodal throughputs at nodes 1 and 2, respectively. D is the distance in slots between the two nodes. δ represents the phase difference in the arrival times at the slots from the two nodes, measured in slot times. Note that if $D=0$ or $\delta=0$, the nodal throughput is evenly distributed

between the two nodes.

It has also been reported in [22] that the same kind of dominance, though to a lesser extent, prevails if there is a group of nodes at light load and a single heavy user bursting in.

1.3.2 Heavy Loading Initiated by a group of users

Next, we consider the case where nodes start quasi simultaneously (within a small time interval) such that a node starts before seeing either busy slots on the forward bus or requests on the reverse bus. In [22, 9, 26], it was reported that in such cases the end nodes hold the largest share of the bandwidth, more or less equally divided amongst them. Intermediate nodes, on the other hand, consume a significantly lower share of the bandwidth. This share increases with increasing speed.

[22, 9, 26] provides a detailed description of why the system behaves in this way. Suffice to say that the unfair behavior is an extension of the effect discussed in the previous section. Intermediate nodes are suppressed by both downstream and upstream stations. Requests from downstream stations forces intermediate nodes to increment their Request counters. On the other hand, segments from upstream stations prevent intermediate nodes from transmitting segments. Therefore, segments transmission from intermediate nodes are delayed further and their throughputs fall below those of

nodes at the ends of the network.

1.3.3 Node Stopping Transmission

In this last case, we shall report the effects of a dominating heavy user stopping transmission. This case was studied and results reported in [22] and it was found that the bandwidth released is equally divided between the two adjacent nodes. The adjacent nodes acquire a considerably greater bandwidth share than the other active nodes. It is plausible to infer that they shared the released bandwidth by the stopping node. The other nodes do not seem to benefit considerably from that node's completion of transmission. Reference [22] provides a detailed explanation of how the bandwidth redistribution happens. The redistribution is attained by the dynamic relation that builds up between the bandwidth released by the node stopping transmission and the adjacent nodes' requests.

1.3.4 Unfair behavior in the priority mechanism

The unfairness problem associated with the DQDB network is persistent even when dealing with priorities [22, 11, 3, 19, 17]. Upstream nodes of lower priority can grab more bandwidth than downstream higher priority nodes, under heavy load. Also, it might happen that changing to higher

priority inadvertently causes low priority initial heavy user to get even more bandwidth. Next, we shall demonstrate these cases.

Consider the case where we have a 150 km network of 5 nodes that are evenly spaced. Assume that we have node 1 as a low priority dominating heavy user. Furthermore, let us assume that node 4 is the high priority would be heavy user, Figure 1.6. Despite node 4 being a higher priority user, it cannot attain its fair share of the bandwidth. Since the bus is already flooded with segments from node 1 when node 4 starts, node 4 is inhibited from transmitting which in turn inhibits requests from being issued on the reverse bus. The only time node 1 lets an empty slot go by is when it identifies a request of a higher priority. It increments its Countdown counter (rather than its Request counter) by one which enforces node 1 to let one slot pass by untouched. This clearly demonstrates that node 4's throughput is dependent on the round trip propagation delay between these two nodes.

In addition to the aforementioned problems, it may also happen that when a node changes its priority level from low to high, the node witnesses a decrease in its throughput. This phenomenon is called the inverse priority effect which is caused by the bunching effect [22]. Bunching effect usually happens when heavy users start quasi simultaneously and it so happens that the free slots on one bus and the requests on the reverse bus forms a special time relationship. This allows the low priority heavy user node closer to the high priority heavy user node to transmit more slots in a batch mode at the

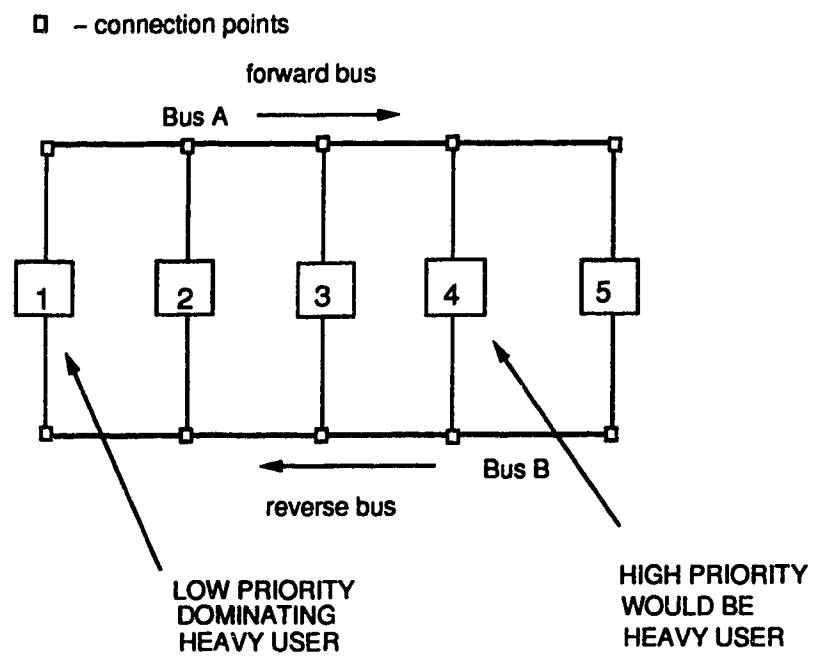


Figure 1.6: Nodes of different priorities

expense of the high priority node.

All the scenarios and investigations presented clearly show that the basic DQDB protocol is unfair under heavy load. Since bursty traffic is unpredictable, it is hard to anticipate the behavior of the node. The unfairness seems to become more profound as the network speed or the network size increases, because the throughput dominant nodes can exert a greater domination as there are more slots available to them or there are more slots they can take from the vulnerable nodes.

1.3.5 Bandwidth Balancing Mechanism

In this procedure, a node is allowed to transmit only a fraction, α , of the time before it artificially increments the Request counter by one for every M segments that it transmits. This M is referred to as the BWB modulus and its value influences the system behavior. The increment operation is achieved by using an additional counter called the trigger counter which keeps track of the number of segments transmitted. As reported in [10], this scheme does prevent throughput dominance by a single user when other users have heavy load traffic demands. However, a fraction of the bandwidth is lost due to passing extra empty slots downstream, which may not be necessarily used. The fraction of bandwidth loss is given by $(1 - \alpha)/(1 + \alpha)$, where $0 \leq \alpha < 1$. A small value for α means that more bandwidth will

be lost. On the other hand, a greater value of α results in less bandwidth wastage, but at the same time slows down convergence to a fair bandwidth distribution among nodes. That is, as the bandwidth balancing modulus M or α increases, the steady state nodal throughputs increase, but it takes longer to reach the steady state [13]. Reference [13] contains an example with 2 nodes separated by a distance D with $\alpha = 0.9$ (the node will transmit 9 segments out of every 10 times that its Countdown Counter reaches zero). In that example, it takes $22D$ slots time for the throughput to reach the 90% steady state throughput. It was also reported that the convergence to a fair distribution of the bandwidth among nodes depends on α only. The mechanism can still maintain unfair behavior owing to long transients needed to achieve balancing [21, 8, 21, 13]. However, the BWB mechanism provides a definite improvement over the basic DQDB protocol [10]. This is achieved at the expense of losing some bandwidth. In fact, wasting a small fraction of the bandwidth allows implicit coordination among the nodes. The BWB mechanism is in a way a rate control procedure.

Bandwidth balancing was originally proposed for single priority traffic. In [12] modifications to adapt the BWB mechanism to multi-priority traffic is discussed. The three modifications that are described in [12] are bandwidth balancing over nodes using local priority information, bandwidth balancing over parcels using local priority information and finally bandwidth balancing over parcels using global priority information. In this case local

information means that nodes will transmit segments of priority \mathbf{P} using a modulus $M_{\mathbf{P}}$ based on the traffic load of that priority level, whereas in global information, nodes will transmit segments of priority \mathbf{P} using a modulus of M . Among these, bandwidth balancing over parcels using global priority information is the most effective and it will be the only scheme presented here. Unfortunately, Global priority information does have more communication and computation overhead than the others. In this scheme, every node can determine the bus utilization due to traffic of each priority level. Each parcel is forced to limit its throughput to some multiple M of the spare bus capacity not used by parcels of equal or greater priority. A node with less demand can have all the bandwidth they desire. The IEEE 802.6 committee has not yet adopted any scheme for multi- priority operation and this issue is therefore still an open area of research.

1.4 Other Proposals

In this section, new proposals based on DQDB for media access protocols for Metropolitan Area Networks will be discussed. As it is not possible to review all proposals, since there are too many of them, a sample of the more prominent and relevant ones are included in this section. All these new protocols have one basic set of objectives. They would like to achieve:

- full bandwidth utilization,
- fairness among nodes,
- minimum possible access delay, and
- efficient priority implementation.

1.4.1 Distributed Queue Multiple Access (DQMA)

Distributed Queue Multiple Access (DQMA) protocol is a generalization of the DQDB protocol. In the DQDB protocol, a node can enqueue at most one request in one slot, whereas in DQMA, a node can enqueue several requests in the same slot. In order to do so, DQMA has an access control field of two bytes out of which eight bits are for requests. Therefore, a node can transmit up to 255 requests. When a node has a packet to transmit, it has to send requests on one bus and segments on the opposite bus. Requests can be sent only if the request field of a slot is empty, otherwise the request transmission is delayed until such a slot is found. After sending a request, a node will enqueue the *local request* in its request queue. This queue also holds *external requests*, which are enqueued if the node identifies a slot with a nonzero request field. The request queue is served on a FIFO basis. If an *external request* is at the top of the queue, then every time an empty slot passes by, the *external request* is decremented by one until it reaches zero.

On the other hand, if the *local request* is at the top of the queue, the node uses all empty slots to transmit its segments. It also decrements the *local request* by one for every segment that it transmits. When the request at the top of the queue reaches zero, the request is deleted. Since the average number of requests per slot is greater than one, some constraints are necessary to keep the request bounded. This is achieved by having an access window that monitors the number of request transmissions. In DQMA, although nodes send multiple requests in a single slot, there is still no guarantee that the segments will be transmitted in contiguous slots. In order to obtain contiguous transmission, DQMA has a request delay equalization scheme [28]. In short, every *external or local request* which is read in is delayed by at least one round trip propagation delay with the headend before they are moved to the request queue. This enables nodes to make contiguous transmissions.

As reported in [28], this protocol enables the system to achieve fairness in throughput distribution even at high speeds and large distances. Reservation of multiple consecutive slots is a new concept introduced by this kind of network and makes segment labelling unnecessary. It also facilitates reassembly significantly as segments from the same node are received consecutively. To provide contiguous slots, it was necessary for nodes to have consistent view of the state of the network. This was achieved at the cost of introducing a delay before *local request* and *external request* before they can be moved to the request queue. Since the delay has to be at least one round trip

propagation delay, this means that the minimum access delay for a segment would have to be at least one round trip propagation delay too. For some types of data traffic, e.g., voice and video, such a delay could be detrimental to the applications. DQMA can be extended to enable segments for immediate access. However, that will leave the system with no guarantee of transmitting slots consecutively.

1.4.2 Cyclic Reservation Multiple Access protocol

Cyclic Reservation Multiple Access (CRMA) is yet another scheme for a metropolitan area network based on slotted unidirectional bus structure. The bus can be either a dual or a folded one. The 2 bytes access control field contains a 13 bit access command subfield. Out of the remaining three bits two are unused and one is used to indicate the status (busy/free) of the slot. There are two main access commands: *reserve* and *start*. The reserve command contains cycle number and cycle length as its arguments. The headend generates periodically *reserve* commands by setting the cycle length to zero. In this scheme a node can reserve multiple slots. This is attained by increasing the cycle length by the number of segments a node intends to send. A node also keeps track of the cycles it has requested to transmit by placing such cycle numbers in its local queues. It is to be noted that a *re-*

serve command is not considered successful until confirmed by the headend. In order to provide fair access to all nodes, a limit is imposed on the number of slots a node can reserve in a cycle [28, 15]. When the headend receives the *reserve* command back, it adds the cycle length to a global reservation queue which is served according to a FIFO discipline. The headend station serves every reservation by issuing a *start* command containing a cycle number, which indicates the start of such a cycle, followed by as many empty slots as requested by the stations for transmission in that cycle. When active nodes observe a start command, they check the local queue for a matching cycle number. If it is a matching one, segments are sent consecutively. Otherwise, transmission is delayed until a *start* with a matching cycle number is observed. In order to monitor the total number of slots in the cycles queued at the headend, including the slots in the outgoing cycle which have not left the headend, a back pressure mechanism is employed [28, 15]. As explained in [28, 15], if the cycle length in the global queue exceeds the round trip propagation delay in slots, the generation of *reserve* commands is inhibited. Furthermore, all reserve commands which have been issued but which have not yet returned are cancelled. The reserve generation is resumed when the number of reserved slots drops below the threshold.

In order to implement p priorities, the CRMA access protocol, both at the headend and nodes, must be replicated p times. These protocols run in parallel. An access command has precedence over another if it belongs to

a higher priority. Also, a cycle can preempt any other cycle of lower priority.

CRMA, just like DQMA, provides throughput fairness even at high speed and large distances. In addition, it also allows transmissions of contiguous segments. This slot contiguity property facilitates packet reassembly. When a folded bus is used, then DQMA has a lower access delay than CRMA. This is because CRMA has a *reserve* command to travel to the folded end of the bus, back to the headend. Then slots must be issued by the headend to the node. Recall that for DQMA, the requests are transmitted on the reverse bus immediately. Therefore, in the case of DQMA with synchronous mechanism for contiguous segments transmission, the minimum access delay is equal to the round trip delay between the node and the headend. In any case, the DQDB medium access protocol seems to have a better immediate access delay than either DQMA or CRMA.

1.4.3 Cyclic Reservation Multiple Access II protocol

Cyclic Reservation Multiple Access II (CRMA II) [23, 24] is based on the experience gained from a 1 Gbits/s implementation of CRMA [28, 15]. Slots are accessed through two distinct mechanisms:(1) immediate access of gratis slots and (2) access of previously reserved slots. In both cases, the busy/free flag needs to be free for a slot to be accessible. The gratis access

mechanism, which is simply based on inspecting the busy/free decisions gives fast access and fully exploits spatial reuse of slots. The principle of reservation is as follows. A special node known as the scheduling node issues a *reserve* command by transmitting a start/end delimiter pair back-to-back. Nodes that want to send a request do so by inserting a slot in the passing *reserve* command, thereby increasing its length. The network latency is also enlarged by switching a delay register into the data path. Upon return of the *reserve* command, the scheduler copies the request entries into its memory while the command passes for a second round trip to eliminate the request of the additional network latency. Nodes successively remove their entries. The scheduler analyzes the requests made by the nodes and sends out a *confirmation* command. This command contains a threshold value. A node cannot transmit more than that threshold value in slots. Immediately after sending the *confirm* command, the scheduler starts to mark all passing gratis slots as reserved until the number of such reserved slots corresponds to the number of confirmed requests. Subsequently, the *reserve* command is issued to collect the request entries for the next cycle.

The priority implementation of CRMA II is analogous to what has been described for CRMA. Just like CRMA, CRMA II provides throughput fairness even at higher speeds and longer distances [28, 15]. Since slots are being transmitted contiguously, the reassembly at the receiving end is easier while the segments labelling at the transmitting end is less complex.

However, adding buffers does increase the complexity of the network. Nodes have to send the requests with the *reserve* command, followed by removing the request from the *confirm* command, and only after receiving the *start* command that nodes are allowed to transmit. Obviously, all this adds considerably to the access delay.

1.4.4 Fair Distributed Queue protocol

The Fair Distributed Queue (FDQ) [25] is a more recent medium access protocol for MANs. FDQ is a slotted system based on the unidirectional bus topology. Slots of fixed duration are generated by a headend station. In this section the terms outbound and inbound channels are analogous to a forward and reverse bus with a folded end connecting the busses. In FDQ, each node has the capability of reading from both the outbound and the inbound channels and writing to the outbound channel only. A slot header contains a busy/free bit, and active (A) bit, and an Inactive (I) bit which are all initialized to a value of zero when the slots are generated. When a slot is written by the node, the busy bit is set to one. When a node becomes active it sets the next unset (A) bit, and after it has completed transmission it sets the next unset (I) bit. These bits enable upstream stations to deduce the status of downstream stations. The (A) and (I) bits can only be set

when the slot is in the inbound channel. Nodes keep track of the number of active nodes downstream by an Active Downstream node (AD) counter that operates even when the node has no packets to transmit. The AD counter is incremented by one for every (A) bit detected on the reverse bus while it is decremented by one for every (I) bit observed by the node. When a node becomes active, it sets the first unset (A) bit it encounters to one. At the same time, the contents of its AD counter is transferred to a second counter which is known as the Count Down (CD) counter. The CD counter is decremented by one for every free slot passing by the node on the forward bus. When CD counter reaches zero, a segment from the node can be transmitted in the first free slot. When a node becomes idle, it sets the first free I bit observed on the reverse bus to one.

Priority implementation in FDQ is an extension of the basic protocol. The counters are replicated for each priority level. In addition, there is a different active and inactive bit for each level of priority. The access mechanism is the same with preference given to segments of a higher priority. If a node has an AD counter of higher priority level with a non zero value, it refrains from transmitting and waits for AD counters of higher priority to drop to zero.

It is reported in [25] that this new protocol achieves throughput efficiency independent of the bus length, the transmission speeds and the number of nodes. FDQ allocates equal bandwidth under heavy load to all

active users in a time period less than or equal to the round trip propagation delay without wasting bandwidth. The priority mechanism is effective as the bandwidth is shared fairly among nodes of same priority level. However, there is a delay involved between a high priority message at the top of the local queue and the time it gets transmitted. In the worst case, this delay could be as long as the round trip propagation delay. Bearing this in mind, the delay is still dependent on the size of the network. Another drawback of this protocol is that there is slot wastage with FDQ due to the time it takes for the (I) bit to propagate upstream.

1.5 Summary

Before discussing the main objective of the thesis, we recapitulate the current technological situation. BISDN wants to provide a means by which data from diverse services can be transported in a unified way. The committee in charge has decided that ATM shall be the multiplexing technique. Besides, the IEEE 802.6 has adopted the DQDB protocol as the medium access protocol for MANs. Studies have revealed that DQDB possesses several inherent problems. Unfairness was manifested for unipriority as well as multipriority traffic. To partially overcome these problems, the IEEE 802.6 committee adopted the BWB mechanism to deal with the unfairness problem. The cost was to sacrifice some bandwidth to help coordination among

nodes. Unfortunately, new problems popped up as the balancing convergence is severely influenced by the speed and size of the network. As the network speed and size increase, it takes longer for the network to converge to the anticipated breakdown of bandwidth. The BWB mechanism is therefore ineffective if the convergence takes so long that the condition it intends to remedy disappears before the operation converges to its final steady state. To fulfill the requirements of BISDN, it is vital to have a protocol that is reliable and fair to all the nodes independent of the time the transmission started. More importantly, since BISDN provides services that are delay sensitive, it requires a protocol that can implement a reliable priority mechanism. Unfortunately, the current standard does not fulfill such objectives. The other newer protocols (DQMA, CRMA, CRMA II, FDQ) that were discussed provide solutions to several of the problems encountered by the DQDB protocol. Unfortunately, they have just stopped short of satisfying the entire objective list.

1.6 Thesis Overview

The purpose of this thesis is to propose a new protocol, the High Performance Distributed Queue, (HPDQ). Just like DQDB and the newer protocols, HPDQ is implemented on a slotted unidirectional bus system. This bus topology has been chosen since high speed transmission require optic fibers

as it supports transmission of several Gbits/sec. This bus topology also complies with ATM specifications of bus systems. The principles of operation of HPDQ are fundamentally different from the other protocols proposed. The primary goals of HPDQ is to achieve full bandwidth utilization and to be fair to all the nodes, independent of the bus length, transmission speeds, number of nodes and starting time of transmissions. In implementing priority, HPDQ aims at providing a reliable service in addition of being robust, i.e. HPDQ intends to provide minimum insertion delay for higher priority nodes. It shall also be shown that in the worst case, redistribution is achieved in a round trip propagation delay. The hardware implementation of the switch is a bit more complex as the switch requires some buffering capabilities. However, this is not a severe disadvantage as most implementations of high speed bus networks use such a switch, e.g., DQDB, DQMA, and CRMA I and II.

1.7 Thesis Outline

In Chapter 2, the basic version of the new protocol High Performance Distributed Queue is described together with an enhanced version. In Chapter 3 performance of the basic and enhanced protocols is studied through simulation. In Chapter 4 the priority implementation on HPDQ is considered. Simulation of the priority implementation of HPDQ is reported

in Chapter 5. Chapter 6 concludes the thesis with a summary, remarks and directions for future research.

Chapter 2

MEDIUM ACCESS

PROTOCOL FOR HPDQ

2.1 Objectives

Despite the efforts made in designing and implementing efficient network protocols for operation in the metropolitan environment, it is unfortunate that none of those protocols is problem free. Such problems can be either technological, economical, behavioral or performance related. To solve those problems, further research efforts is needed in this area.

This thesis extends yet another effort whose goal is to design a network protocol that is efficient, cost effective, and above all problem free, by proposing a new protocol: The High Performance Distributed Queue

(HPDQ). This protocol is defined on a dual unidirectional slotted bus system with active taps. Slots are generated at equal intervals by the end nodes. The HPDQ protocol has been designed while bearing in mind the problems that were encountered by previous protocols defined on a similar bus topology. The protocol reduces and in fact eliminates most of those problems.

To be efficient, a protocol must make use of the total bus capacity. In order to prevent any node or group of nodes to take control of the network, a protocol must provide fairness among nodes. These are two of the objectives of the HPDQ protocol. In addition, the HPDQ protocol intends to provide the minimum access delay to segments. Since minimizing bandwidth wastage is another important issue considered in numerous protocols, the HPDQ protocol made that issue an additional goal. Next we describe how the HPDQ protocol attains these goals.

2.2 Description of HPDQ

In this section the basic HPDQ protocol for unipriority operation is described and will be extended later to multipriority. Although this protocol suffers from a certain deficiency that will be remedied by the enhanced version, it is described in order to explain the principle of operation.

Since HPDQ operates on two unidirectional busses serving two oppo-

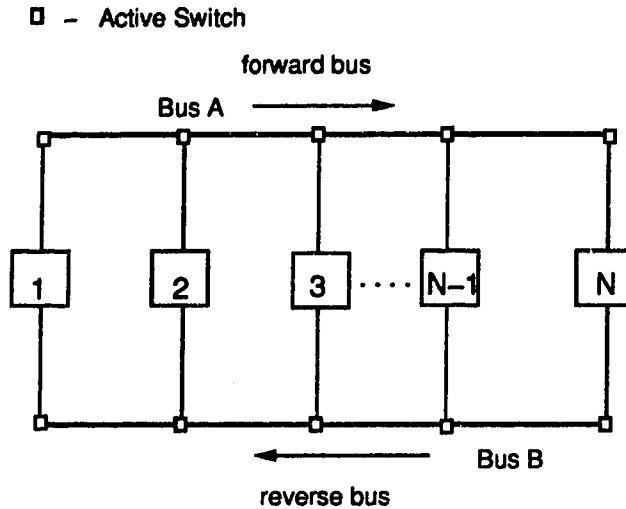


Figure 2.1: HPDQ network configuration.

site directions and, since protocols for both busses are identical, the protocol for bus A only will be described. Bus A will be referred to as the forward bus while bus B will be called the reverse bus. The nodes are tapped onto the busses through active switches, see Figure 2.1. Through such switches, nodes can buffer slots and modify them with a minimum delay and then release them back onto the bus. Just like several other protocols defined on this bus topology, the HPDQ protocol is implemented through the use of counters. A node will have two basic counters: the number of Active Nodes Downstream counter, *AND_ctr* and the Count Down counter, *CD_ctr*. The finite state machine of the basic protocol is shown in Figure 2.2. The *AND_ctr* keeps track of the number of nodes that are active downstream. It is to be noted that the *AND_ctr* is not essential for the operation of this

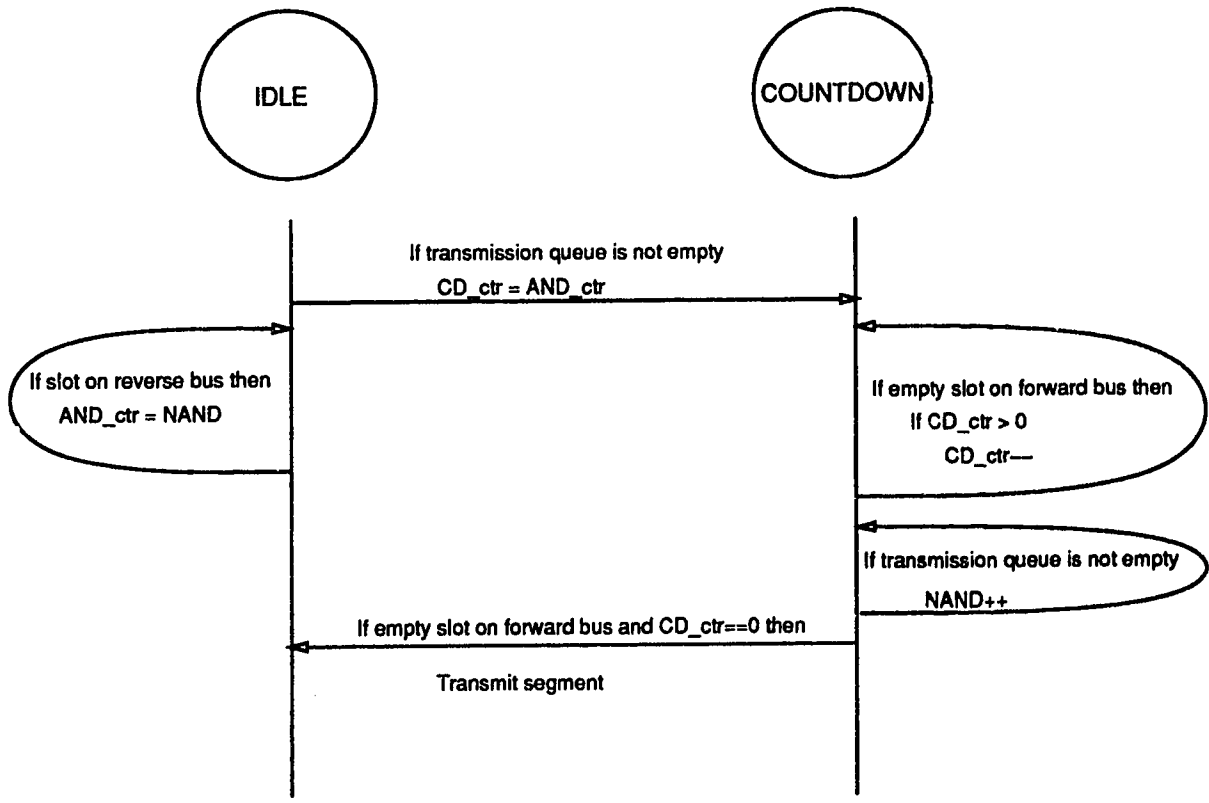


Figure 2.2: Finite State Machine for Basic HPDQ protocol.

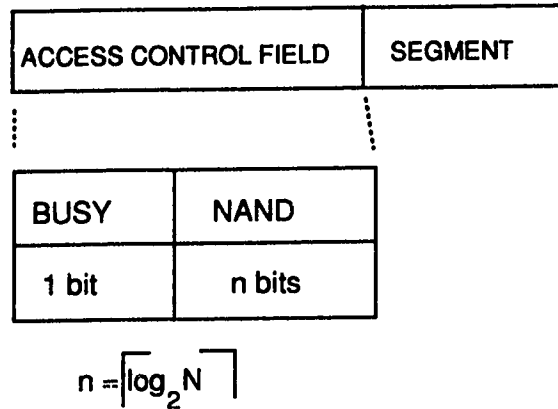


Figure 2.3: Slot format for basic HPDQ.

protocol and its sole purpose is for ease of understanding of the protocol. In fact the *AND_ctr* is numerically equal to the value in the NAND field which shall be described later. The *CD_ctr* holds the number of unused slots that the node should let go before consuming one. If the *CD_ctr* is equal to zero and the transmission queue is not empty, the node transmits in the next empty slot. Then, the *CD_ctr* is used in a manner similar to that in the DQDB protocol. This protocol relies on the reverse bus to provide information about the status of the nodes downstream. In order to do so, the slots need a field which will enable the upstream nodes to know the number of active stations downstream. We shall refer to this field as the NAND (Number of Active Nodes Downstream) field and the slot format is given in Figure 2.3. When a station has segments to transmit, it will increment the value of the NAND field of the next slot on the reverse bus by one. This can be

implemented with a single bit delay per node. The NAND field is updated as long as the transmission queue is not empty. Since the value contained in the received NAND indicates the number of active nodes downstream, its value is copied to the *AND_ctr*. The minimum length of the NAND field is $\lceil \log_2(N) \rceil$ bits, where N is the number of stations on the bus. Since a typical MAN nowadays can contain up to 1000 stations, it shall be assumed that N is 1000. Therefore, a NAND field of 10 bits will suffice for a MAN.

In this protocol, nodes copy the *AND_ctrs* to the *CD_ctrs* everytime the *CD_ctr* reaches zero. Therefore, *CD_ctrs* hold the number of slots required by downstream stations. The *CD_ctr* is decremented by one for every empty slot that a node observes on the forward bus. When the *CD_ctr* is equal to zero, the node transmits its segments into the next empty slot. The *CD_ctr* is then reinitialized to the new *AND_ctr* value.

To illustrate the operation of the HPDQ protocol, consider the following example where there are 5 nodes out of which only nodes 1, 2 and 4 are active. The internode distance is 5 slots. Figure 2.4 shows a snap shot of the contents of the NAND fields, the *AND_ctrs* and the *CD_ctrs* of the five nodes. Notice that the value of the *CD_ctrs* of nodes 3 and 5 are zero. However, since they are not active, they will not transmit. Nodes 1 and 3 will transmit segments after letting two empty slots and one empty slot go by, respectively. Since the *AND_ctr* of node 4 is zero, it will transmit in every empty slot.

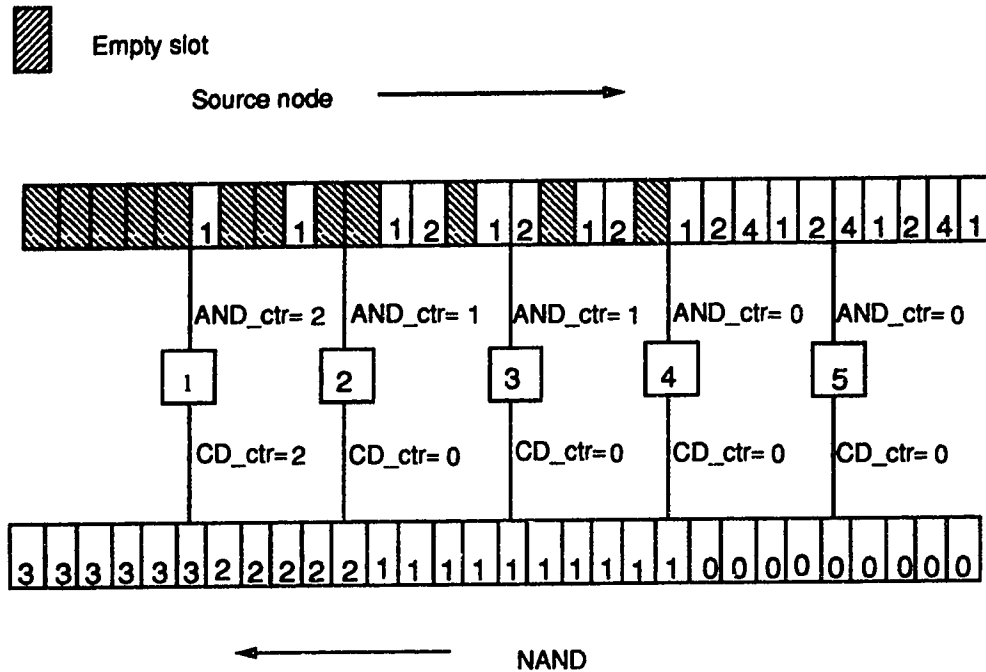


Figure 2.4: Basic operation of HPDQ

The protocol as it stands might attain 100% channel utilization when a group of stations are active for a long time. Unfortunately, HPDQ cannot maintain this level of utilization when stations start switching between the active and inactive states frequently. This is due to the effect of propagation delay. To illustrate this problem, the transient behavior of 2 nodes that are 20 slots apart is studied. The network configuration is shown in Figure 2.5.

Upon completion of its transmission, node 2 stops updating the NAND field on the reverse bus. This means that the next 20 slots that node 1 will read in, will have a value of 1 in the NAND field. Node 1 will therefore

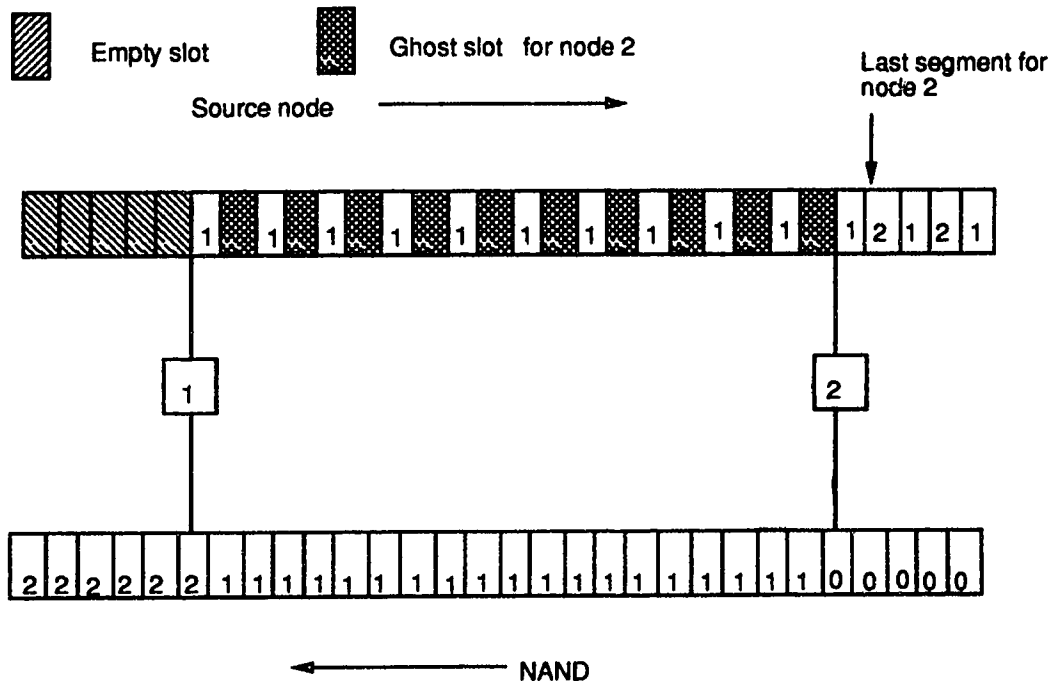


Figure 2.5: Node 2 stops updating at completion of its transmission

assume that node 2 still requires empty slots to complete its transmission. This is clearly an erroneous interpretation. Hence, node 1 will release ghost slots which will not be consumed by node 2. Therefore, bandwidth is lost which leads to a transient wastage of the bandwidth when utilization drops below the 100% level while some nodes are still active. The transient behavior of the node in this example is shown in Figure 2.6. Note that time is measured in slots. Node 2 terminates its transmission at 520. Node 1 acknowledges that at 540 by consuming all the empty slots, thus reaching a utilization of 1.0. In the transient period between 520 and 540, node 1 was allowed a maximum utilization of only 0.5 although the other 50% of the bandwidth was wasted. It should be pointed out that, in general, bandwidth is wasted only when the most active downstream station completes its transmission.

Due to this, a modification to the original HPDQ protocol to overcome the above mentioned problem will be introduced next.

2.3 The Enhanced HPDQ Protocol

As has been illustrated in Section 2.2, the completion of transmission by the most downstream active node results in a possible loss of bandwidth due to the effect of the propagation delay. This can be reduced, if not elimi-

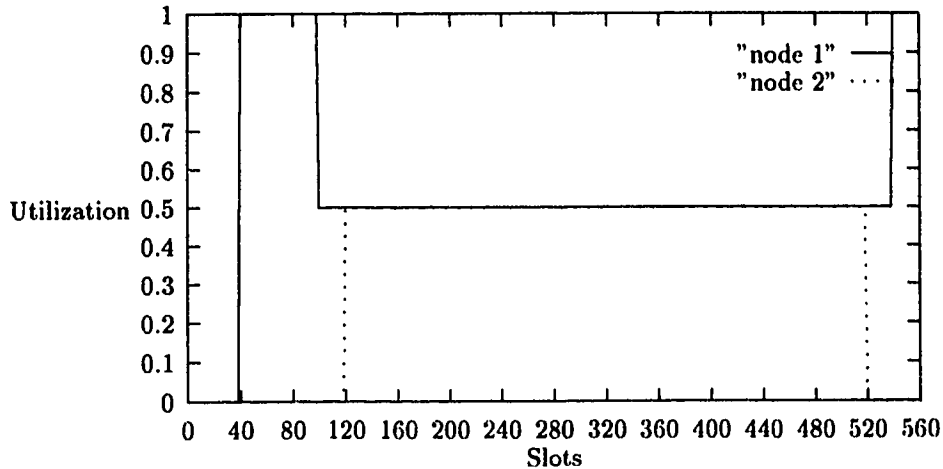
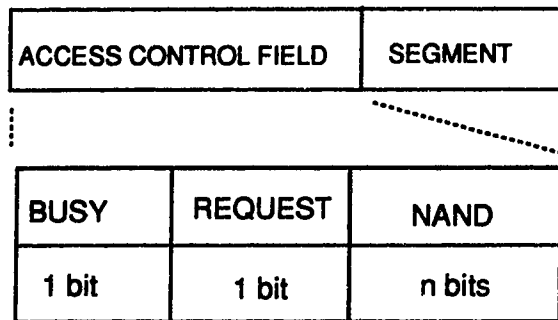


Figure 2.6: Bandwidth loss due to stopping of transmission.

nated through the use of a special counter.

To implement the enhanced version of the HPDQ protocol, each node needs to have an additional counter, the Cumulative Request counter (CumReq_ctr) and each slot should contain an additional field, namely the Request bit field shown in Figure 2.7. The CumReq_ctr keeps track of the number of requests needed by downstream stations. The Request bit field differs from the NAND field in that, the former conveys requests from active stations whereas the NAND field indicates the number of nodes that are active downstream. The Request bit is set as it passes through an active node with pending requests, on a per segment basis. If the incoming Request field is already set then the node has to wait for the next available Request



$$n = \lceil \log_2 N \rceil$$

Figure 2.7: Slot format for enhanced HPDQ.

field. When a node has sent all its requests, it stops setting the Request field. However, it updates the NAND field as long as the segment transmission is incomplete. The finite state machine is shown in Figure 2.8. The CumReq_ctr is decremented by one for every empty slot that the node lets go by on the forward bus. It is incremented by one for any requests arriving on the reverse bus. In fact, as long as CumReq_ctr is greater than zero, the node will pass a number of empty slots that is equivalent to the value of the NAND field for every segment that the node transmits. When the CumReq_ctr of a node reaches zero, the CD_ctr is set to zero and that node will consume all empty slots on the forward bus regardless of the value of the NAND field. A value of zero in the CumReq_ctr implies that stations downstream have completed all transmissions. As active nodes with a CumReq_ctr of zero will consume all empty slots, this means that no bandwidth is lost.

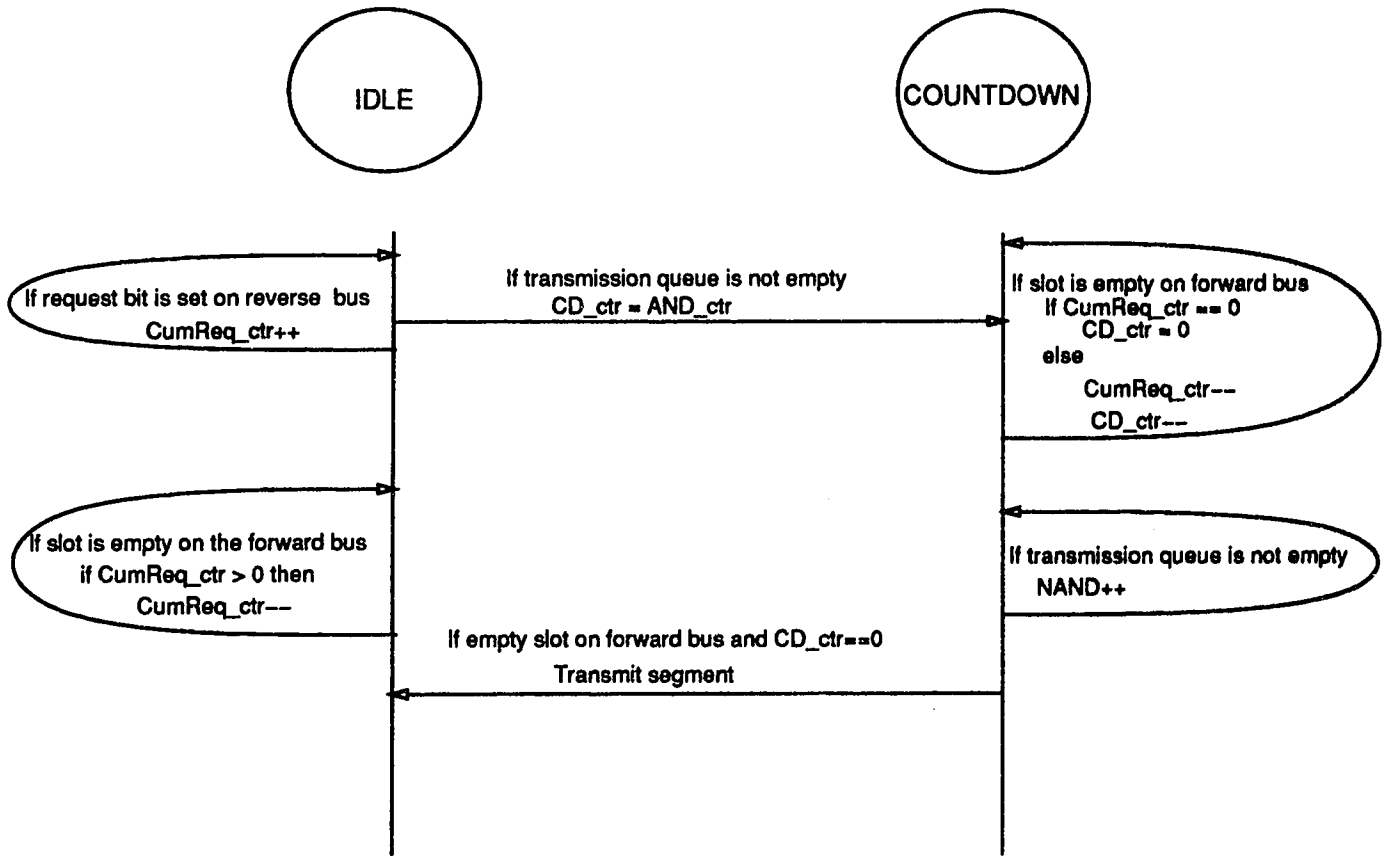


Figure 2.8: Finite State Machine of Enhanced HPDQ protocol.

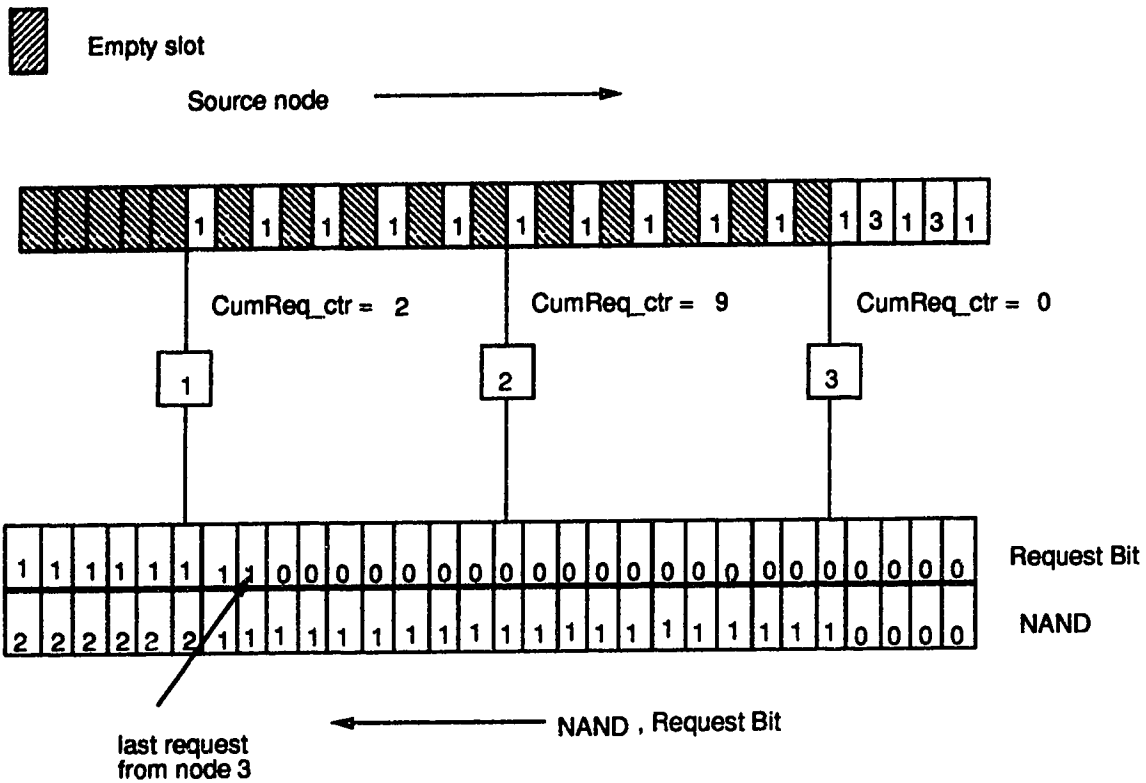


Figure 2.9: Operation of the CumReq_ctr counters.

Figure 2.9 illustrates the outcome of this approach. The 3 nodes numbered 1 to 3 are 10 slots apart. Only nodes 1 and 3 are active. Node 1 has its CumReq_ctr with a value of 2 in it, whereas node 3 has a 0 in its CumReq_ctr. At the next slot time, node 1 will pass an empty slot downstream. Meanwhile, it will read the request from node 3 on the reverse bus. Since its CD_ctr will be equal to zero at this instant, it will transmit in the next free slot. At the end of this transmission the CumReq_ctr would have

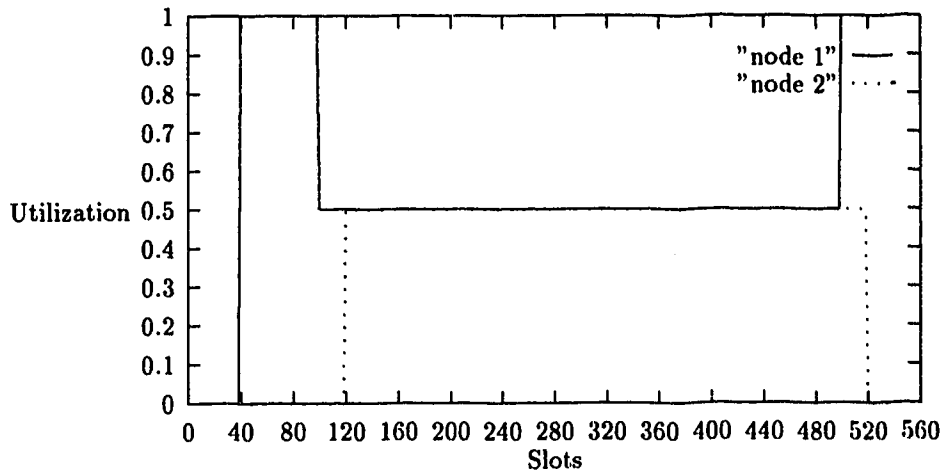


Figure 2.10: Bandwidth distribution due to node stopping transmission.

been updated to a value of 3. The CumReq_ctr of node 1 will not increase any more since there are no more requests from node 3 on the reverse bus. For the following six slots, node 1 will use every other slot and let an empty slot propagate downstream to node 3. Afterwards it will consume all free slots on the forward bus.

In order to understand the gain brought about by the introduction of the Cumulative Request counter, let us examine the scenario as described in Figure 2.5 but with the new set of rules.

Node 2 updates the Request field on the reverse bus for every segment that it has to transmit. The CumReq_ctr of node 1 is incremented by one for every Request field that node 2 has modified. For every empty

slot that node 1 receives on the forward bus, node 1 reinitializes the *CD_ctr* to 1, as the *NANL_ctr* is 1, and decrements the value of the *CumReq_ctr* by 1. Since *CumReq_ctr* keeps track of the exact number of slots that the downstream node require, there is no need to let any more empty slots pass by once the *CumReq_ctr* value reaches zero. Consequently, node 1 will consume all incoming slots. This enables the system to reduce, if not eliminate all sources of bandwidth wastage. The transient utilization shown in from Figure 2.8 clearly illustrates the performance gain that the enhanced HPDQ protocol provides. The total transient utilization stays at 100% and there are no more ghost slots.

2.4 Network Aspects

In this section we shall discuss the performance and implementation aspects of this new protocol. This will be done in order to gain an insight into the protocol. Therefore, no comparison to other networks will be made.

2.4.1 Performance issues

In order to investigate the effects of the HPDQ protocol on performance, let us consider an HPDQ network with N nodes. For the sake of this

analysis, assume that all active nodes have a load that is much greater than one. In other words, the transmission queue is never empty. Let N^* be the number of active nodes, where $N^* \leq N$ and let X be one of the N^* active nodes. Let us further assume that there are N_1 active stations upstream from X and N_2 active stations downstream from X , such that $N_1 + N_2 = N^* - 1$. Once the system has stabilized, node X consumes one slot out of every N^* slots. Therefore an active node, X , will acquire $1/N^*$ of the available bandwidth and an average insertion delay of $N^* - 1$ slots, for $N^* > 0$.

Under the above conditions, the bus utilization reaches 100%. This is because when the system is active and stable, every slot on the bus is a dedicated slot. The bus utilization is independent of the actual number of active stations. It depends only on the load that the nodes offer to the system, and will reach 100% if there are enough segments queued up.

This new protocol is virtually insensitive to nodes leaving and entering the system and to their relative positions. At two different time instants, t_i and t_{i+n} , the system can have 2 different sets of N^* active nodes. Several stations of the first group of active nodes may have terminated their transmissions and have been replaced by another group of new nodes that become active. When this happens and the system has stabilized, the average bus utilization and average access delay of individual nodes remain the same. This is because the delay is based on the number of active nodes, regardless of the location of these nodes on the bus. As shown earlier, if we have N^*

active nodes with an infinite number of segments queued up for transmission, nodes will still transmit in no more than one slot out of every N^* slots that go by, such that the throughput is $1/N^*$.

The fluctuation in the number of active stations does not affect the average bus utilization as long as there are enough segments to keep the bus busy. However, the average access delay is highly dependent on the number of active stations. A drop in the number of active stations enables the remaining nodes to transmit segments more frequently and consequently reduces the average access time. Similarly, if the number of stations increases the average access time will increase too since every node will have to let more slots go by before consuming one.

2.4.2 Implementation Cost and Complexity

The implementation cost and complexity are very important issues to consider if we want to implement a protocol. The HPDQ protocol was designed by giving special considerations to such issues. The performance gains can be regarded as satisfactory. The enhanced HPDQ protocol will be the only one considered here since it has all the qualities of the basic HPDQ protocol together with some additional attractive features.

Counters

In this protocol, for each bus, a node has one *Active Number of Downstream nodes* counter, *AND_ctr*, a *CountDown* counter, *CD_ctr* and a *Cumulative Requests Counter*, *CumReq_ctr*. All of these can be implemented in software. Certainly, it will be faster if implemented by dedicated hardware. The size of the AND and CD counters has to be at least as large as the number of nodes that a Metropolitan Area Network can support. *CumReq_ctr* should be large enough to hold requests coming from downstream stations and its size will be dictated by the number of segments that downstream stations intend to send. In order to prevent the *CumReq_ctr* from overflowing, a mechanism that inhibits nodes from transmitting excessive requests is welcome. A back pressure mechanism is a reasonable choice. The headend could monitor the number of requests sent by the nodes. It will keep track of the number of request it receives and if they exceed a certain threshold value, the headend issues a special *stop* command demanding stations to stop request transmission until further notice. When the number of requests has drop below the threshold value, a special *resume* command could be issued to notify stations to resume request transmissions. These commands could be represented by an additional bit in the Access Control Field.

Switches

As the HPDQ protocol makes use of active switches, we should add the cost of these to the implementation cost. The cost of implementing counters is considerably less than that of the switches. The special function which is performed by switches is the modification of the content of the NAND fields. Updating of the NAND field can be done on a bit by bit basis.

2.5 Summary

In this chapter, a new protocol HPDQ (High Performance Distributed Queue) defined on dual unidirectional bus topology, has been described. It provides good performance results and overcomes the problems that other protocols defined on similar bus topologies encountered. The basic protocol has been discussed which was later modified to the enhanced version. The main difference between the basic and the enhanced version is that when the most active downstream node leaves the system, the bandwidth wastage is reduced to a minimum in the enhanced version.

Chapter 3

PERFORMANCE ASPECTS

In this chapter the performance of the basic and enhanced versions of the HPDQ protocol is studied. The average bus utilization and the average access delay to the bus will be our performance measures. With the average bus utilization we can get a better understanding of how the bandwidth is divided among nodes under various conditions. Similarly, the average access delay to the bus enables us to analyze the average length of time a node has to wait before completing transmission of a segment under various conditions.

A simulation model for the HPDQ protocol was considered more appropriate than a theoretical model. The main reason being the complexity of a theoretical model and its analysis. The performance results contained herein are based on simulations of several scenarios. First, we discuss the

steady state analysis of the basic protocol, followed by the transient analysis. It is to be noted that under steady state and heavy load, either version of the protocol will yield the same results. Therefore steady state analysis of the enhanced version of HPDQ shall not be considered. However, the transient behavior of nodes leaving the system shall be analyzed for both versions.

In order to test the goals of our basic and enhanced protocol, a simulator that models the behavior of both protocols was implemented in software. The simulator was designed in such a way that it is flexible and the relevant parameters could be modified easily. Such parameters are: the number of nodes on the bus, the internode distance, the time at which transmission should start and stop, the packet arrival process per node, the distribution of packet lengths, the slot duration, the message and slot overhead and lastly the phase difference in slot generation on both busses. In order to be able to compare the HPDQ to the protocols proposed in the literature, we have chosen our message arrival process to follow a Poisson arrival. The start and stop time of transmissions were controlled whenever the transient behavior of the nodes was of interest.

3.1 Simulation of the Basic Protocol

3.1.1 Steady state analysis

In this section, it is shown that the basic HPDQ protocol divides the bandwidth fairly among all nodes independent of the time instants at which they started transmission. Moreover, this is done without bandwidth wastage. It will also be shown that this protocol has a reasonable average access delay. Where relevant, the drawbacks of the basic HPDQ will be brought up. It will also be shown that convergence is achieved within the round trip propagation delay.

In the first scenario, the network has five active nodes. We shall assume that the transmission queue is always nonempty - a condition which is sometimes referred to as nonelastic traffic. The nodes start transmission at random times. The distances between the nodes were varied to represent transmissions at the speeds shown in Table 3.1 and nodes are equally spaced on the bus. As expected, when the system stabilizes, the results are exactly the same for the different speeds. Table 3.2 summarizes the results for different bus loads that are evenly distributed among all nodes. From Table 3.2 we can see that the bandwidth is equally and fairly distributed among all active nodes under all loads. In fact, under steady state and heavy load, it does not make any difference whether it is the basic or enhanced version of the protocol.

BUS SPEED	DISTANCE OCCUPIED BY SLOT(53 OCTETS)
44.7 Mb/s	1896 m
155.5 Mb/s	546 m
622 Mb/s	137 m
1244 Mb/s	69 m

Table 3.1: Maximum bus length for throughput fairness

Bus Load	Utilization				
	1	2	3	4	5
50	0.10	0.10	0.10	0.10	0.10
60	0.12	0.12	0.12	0.12	0.12
70	0.14	0.14	0.14	0.14	0.14
80	0.16	0.16	0.16	0.16	0.16
90	0.18	0.18	0.18	0.18	0.18
100	0.20	0.20	0.20	0.20	0.20
110	0.20	0.20	0.20	0.20	0.20

Table 3.2: Node utilization.

In our second scenario, a group of nodes start transmitting earlier from the rest of the nodes. The objective here is to see how the nodes adapt to the addition of a new node. In order to investigate the behavior of the protocol under the above mentioned conditions, a simulation was run with 5 nodes that are 10 slots apart from each other. The utilization and the access delay of the nodes for various bus loads was then recorded when steady state was reached. The results were the same independent of the position of

Bus Load	Average Access Delay (slots)				
	1	2	3	4	5
50	36.2	37.1	38.2	39.1	39.4
60	42.1	42.4	43.5	44.8	44.8
70	46.2	50.4	50.6	51.6	51.5
80	59.3	61.1	61.6	62.7	62.8
90	75.5	76.1	75.5	76.7	77.5
100	119.7	119.5	119.5	119.6	119.6
110	119.7	119.6	119.6	119.9	119.9

Table 3.3: Access Delay

the new node and match those of Table 3.2. for utilization and Table 3.3. summarizes the results for average access delay at various loads. Table 3.2 shows that the nodes have the same utilization for a given bus load, whereas Table 3.3 illustrates that the average access delay of the nodes are not too far apart from each other.

In order to investigate the behavior of our system new heavy users are added to a system, simulation results were collected from a network operating with the transmission for the speeds shown in Table 3.1. The network starts with one heavily loaded node, and the four other nodes are gradually added to the system. The results were again the same for all speeds. The utilization of the nodes after the system had stabilized was evenly distributed among all nodes. The results are summarized in Table 3.4. As can be observed, the system remains fair to all the nodes and they get equal shares of

Initial node	Utilization				
	0	1	2	3	4
0	0.2	0.2	0.2	0.2	0.2
1	0.2	0.2	0.2	0.2	0.2
2	0.2	0.2	0.2	0.2	0.2
3	0.2	0.2	0.2	0.2	0.2
4	0.2	0.2	0.2	0.2	0.2

Table 3.4: Utilization for different initial heavy users.

the bandwidth.

In the scenarios discussed above, the results were obtained under steady state and after the system has stabilized. The stabilization process itself could take a time that is long enough for nodes to conclude their transmissions. In this case the protocol fails to provide efficient service if it responds to changes very slowly. Next, we study the transient behavior of the HPDQ protocol in order to investigate its responsive property.

3.1.2 Transient analysis

We shall demonstrate that with the basic HPDQ protocol, the network stabilizes within a reasonable time interval. Special considerations shall be given to the mean access delay.

We shall consider an example in which the network consists of 5 nodes that are equally spaced on the bus and are 10 slots apart. All nodes are inactive and start becoming active one at a time, starting from the most upstream

station. Table 3.5 shows the time(in slots) at which messages arrive to various nodes. The transient results were recorded and are summarized in Figure 3.1. Figure 3.1 shows that it takes a short time 20 slot time for a node to gain access to the bus and a full round trip propagation delay with the most active upstream station for the system to attain stability. To understand how bandwidth is assigned, we trace the life cycle of some of these nodes until they start transmitting. Node 2 enters the system at time 640. At time 660 the utilization distribution is as shown in Table 3.6, where node 1 shares its bandwidth equally with node 2. Since node 0 does not know about the presence of node 2 at first, it still consumes 0.5 of the bandwidth. It will take 20 slots for node 0 to become aware of node 2's start of activity. At this moment, node 1's utilization will immediately drop to $1/3$. Twenty slot times later, the utilization will be equally distributed among all other nodes as shown in Table 3.7. In summary, a new active node has to wait for a total of 20 slots before it can start transmitting. This waiting time is equivalent to the round trip propagation delay between the new node and its immediate upstream active node. The system will stabilize only after the most upstream active station has acknowledged the presence of the new active station. The stations utilization will even out as the empty slots propagate downstream. This means that in the worst case the system will stabilize in one round trip propagation delay.

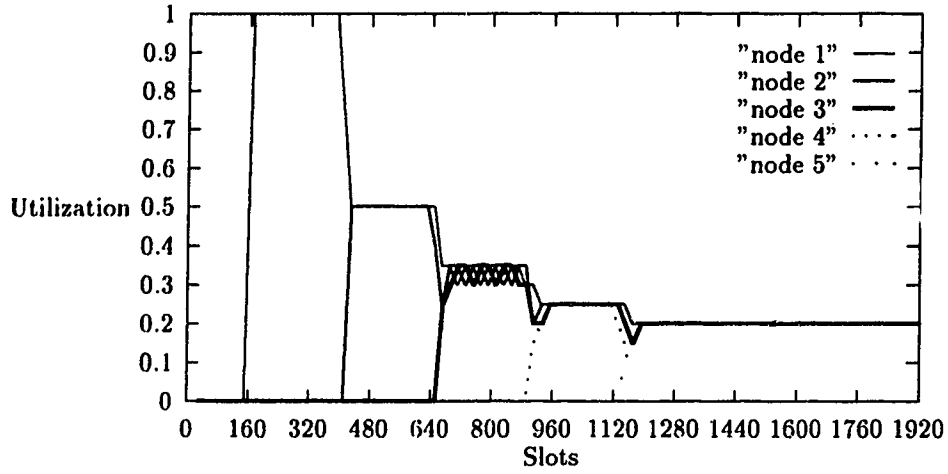


Figure 3.1: Transient utilization of nodes starting from left to right

Node	0	1	2	3	4
Start time (in slots)	160	400	640	880	1120

Table 3.5: Summary of system.

Node	0	1	2
Utilization	0.5	0.25	0.25

Table 3.6: Utilization at 660

Node	0	1	2
Utilization	1/3	1/3	1/3

Table 3.7: Utilization at 680

Node	0	1	2	3	4
Distance (in slot times)	-	50	100	150	200
Start time (in slot times)	1000	2000	3000	4000	5000

Table 3.8: Summary of system.

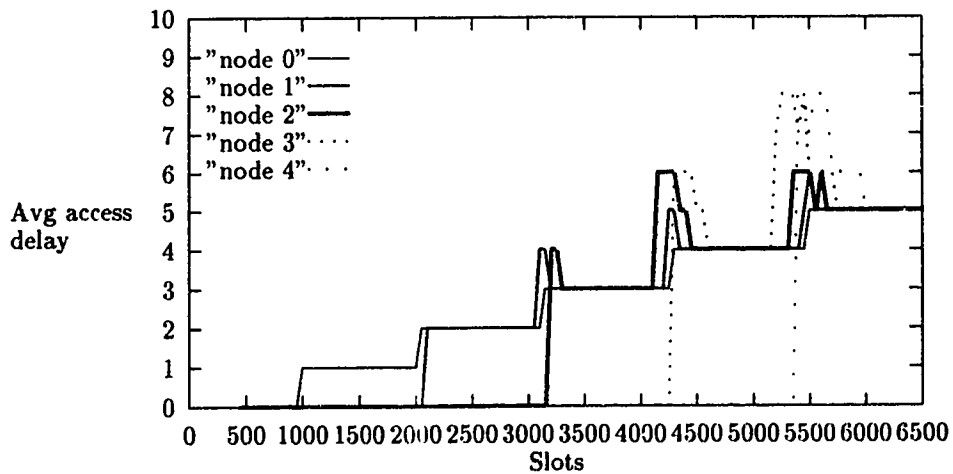


Figure 3.2: Transient behavior of nodes at different distances from each other.

To verify that the access delay is dependent on the distance between the node and its immediate upstream active station, a simulation was run with nodes at various distances from each other. The location of the nodes relative to the previous active node and their starting times are summarized in Table 3.8.

Figure 3.2 shows that nodes that are further apart from the immediate upstream active node have to wait longer to access the bus initially. This

is because nodes need a full round trip propagation delay with the immediate upstream active node before they can start transmission. Since stations do not acknowledge the presence of a new node at the the same time, long waiting time will persist for a while. After a round trip propagation delay with the headend node, the access delay in slots will converge to a steady value that is equal to the number of stations. In addition, Figure 3.2. also shows that as the number of stations increases, so does the access delay.

Although redistribution is attained fairly quickly when a new active node is added to the system, the loss incurred when nodes leave the system is still not justifiable. Under such circumstances, the system loses bandwidth depending on the relative position of the node. To investigate and quantify the loss, a network of 5 nodes with an internodal distance of 100 slots was simulated. Two cases were considered. In both cases the nodes leave the system at 11500 slot times after transmitting 2000 slots. We shall assume that the other nodes are always active. In the first case, nodes 3 leaves the system at 11500 after transmitting 2000 slots. Node 2 acknowledges this change in system only at 11600. The next 40 ghost slots dedicated to node 3 remain untouched as the node is no longer active. However, they are not completely lost as the downstream nodes will end up consuming them. This case is presented in order to compare it to the second case.

When the most downstream station leaves the system at 11500 slot times as shown in Figure 3.3 which depicts the transient behavior of the sec-

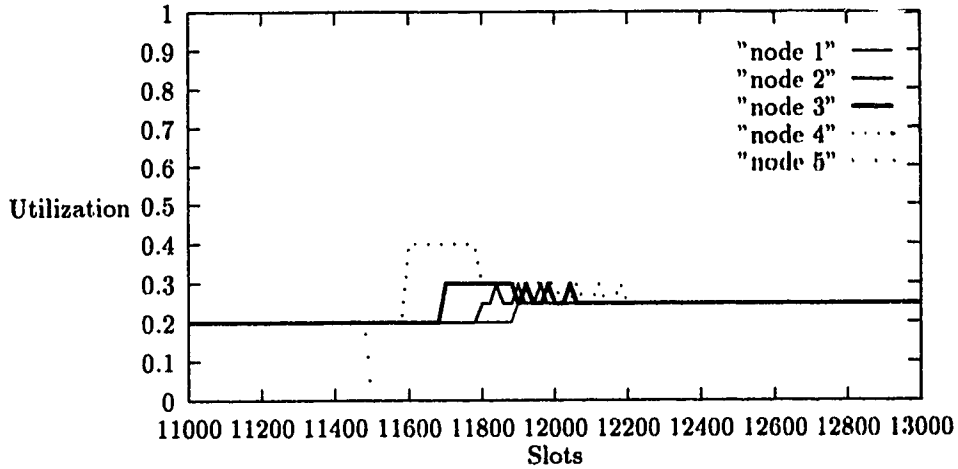


Figure 3.3: Transient utilization of node 5 leaving the system.

ond case, the effect is more profound as slots are lost forever. By the time node 4 acknowledges the completion of node 5's transmission, 200 slots would have already gone by. Unfortunately in this case, the 40 ghost slots left for node 5 are not consumed by any other node. Figure 3.3 shows that node 5's utilization drops at 11500 slot time, at which time the 40 slots are lost forever. This loss will be repeated everytime the most downstream active node completes its transmission.

3.2 Simulation of the Enhanced Protocol

3.2.1 Transient analysis

The enhanced protocol does not change the performance obtained by the basic HPDQ protocol under steady state. Only when the most downstream active node leaves the system that the 2 protocols differ. In the last scenario of the previous section it was shown that bandwidth is lost whenever the end active station completes its transmission. The bandwidth released is not consumed by other nodes. This loss can be overcome through the use of the enhanced protocol. The transient behavior of the nodes was recorded under the same scenario but with nodes implementing the enhanced HPDQ protocol. End results are unchanged when node 3 leaves the system when compared to the previous case. The effect of node 5 leaving the system was different. The upstream nodes know exactly how many empty slots they should pass. After they have provided downstream stations with the number of slots requested, they start consuming all empty slots. Therefore, no bandwidth is lost as shown in Figure 3.4.

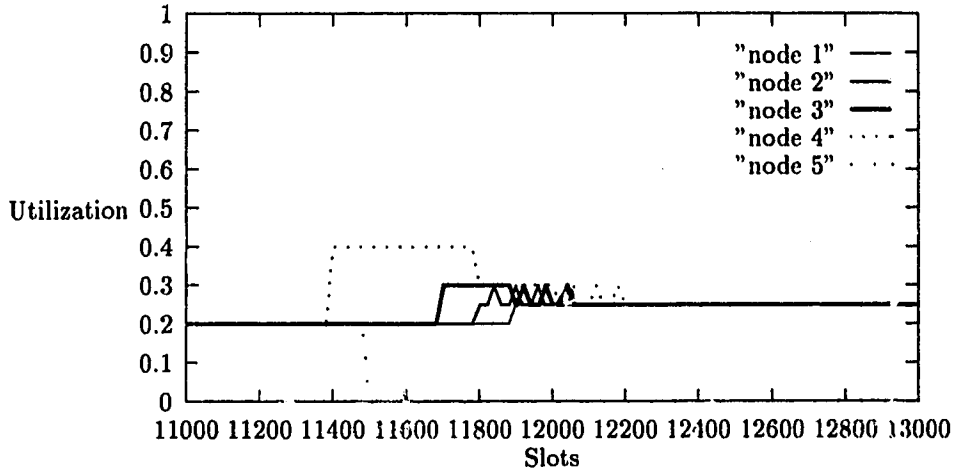


Figure 3.4: Transient utilization of node 5 leaving the system (Enhanced HPDQ).

3.3 SUMMARY

In this chapter, we have analyzed the performance of both versions of the HPDQ protocol using a simulation model. Steady state analysis as well as transient analysis were performed on the HPDQ protocol. Since the enhanced version produces results similar to that of the basic version under steady state, the steady state behavior of the basic version was the only one discussed in this chapter. The steady state analysis showed that the utilization is divided equally among active nodes and the access delay is equal regardless of the time these nodes became active, their locations on the bus and the load offered by the nodes. Also, nodes entering a system with dom-

inating heavy load users were easily integrated into the system. Those new nodes forced the dominating heavy load users to share the bandwidth equally with them and the access delay converged to a value that corresponds to the current number of active nodes.

From the transient analysis of the case where new nodes become active, the results produced herein show that newly active nodes gain access to the bus within a round trip propagation delay from their immediate upstream active node. However, it takes a full round trip propagation delay from the headend node for the utilization and access delay to converge to the steady state value. These results hold for both versions of the HPDQ protocol. It has also been shown that when nodes leave the system, there is some bandwidth loss in the case of the basic HPDQ protocol. The enhanced version remedies this phenomenon by recording the number of slots that downstream users need. In conclusion, the results obtained illustrate the merits of the HPDQ network.

Chapter 4

Priority Implementation

4.1 Introduction and Background

Earlier in this thesis, we listed some of the possible applications of the Broadband Integrated System Digital Network. Some of these applications are delay sensitive. For example, excessive delay in voice packet delivery could adversely affect the quality of the speech playback. Likewise, in the case of video transmission, the inability to deliver frames within certain time bounds could introduce flickering in pictures, thereupon deteriorating the picture quality. It is sensible to infer that some types of data should be given a higher transmission priority over others. In fact, Network Management messages should fall under the highest priority class.

In a packet based non-preemptive priority scheme, segments of lower

priority that are already queued will be transmitted before incoming higher priority segments. A node will differentiate in favor of the higher priority segments when segments are multiplexed into the queue. Therefore, a non-preemptive priority scheme will not be suitable if the incoming segments are delay sensitive.

A preemptive priority class scheme is more advantageous when dealing with delay sensitive traffic. In this class, segments from lower priority packets are preempted until all higher priority segments have been transmitted. This class ensures that higher priority traffic is transmitted at the earliest opportunity. In this case Network Management messages will get the proper attention they require. A preemptive priority scheme is therefore more suitable for multimedia communication.

In [17], two definitions which must be satisfied by all priority schemes are stated:

Definition (1) *If a priority scheme is working then the average access delay (for a given packet size) is substantially less (half or less) for high priority traffic as opposed to the low priority ones.*

Definition (2) *If there is a certain distribution of high priority traffic on the bus, then the access delay experienced by the high priority traffic (for a given size of traffic) should be almost independent at high load of the amount of low priority traffic on the bus. There should be a predictable access delay experienced by the higher priority traffic for a given distribution of traffic.*

Both definitions state that the high priority traffic should get a lower average access delay than the lower priority traffic. However, definition(1) entails that the high priority traffic access delay is heavily dependent on the level of lower priority traffic. Furthermore, in definition(1) the higher priority traffic is not bounded in a predictable way. Definition(2) is more stringent and provides a clearly defined Quality of Service. To conform to definition(2), a preemptive priority scheme at the slot level is needed. Hereafter, we shall use definition (2) for priority.

As specified in definition(2), one of the most important aspects of a priority scheme is how long it takes for the system to adjust to the high priority demand. In the priority scheme implemented in the IEEE MAN Standard, DQDB, the initial problem was its inability, under heavy load, to respond to high priority nodes when an upstream lower priority heavy user node starts transmitting before the downstream ones [22]. A classical example is one in which there are two nodes 50 slots apart with the lower priority node being upstream. If the lower priority node starts more than 100 (the round trip propagation delay between the two nodes) slots before the high priority node, the low priority node will consume about 99 percent of the bandwidth [17]. The bandwidth distribution among the nodes is dependent on the round trip propagation delay between the high priority node and the closest upstream low priority heavy user. In any case, the low priority dominating heavy user is virtually unaffected by the high priority node [22, 17].

The only effect that an incoming request from the high priority node has is that the heavy load low priority node will release a slot immediately regardless of the count in its CountDown Counter. This anomalous behavior is somewhat reduced by using the Bandwidth Balancing (BWB) mechanism. However, as pointed out in [12], BWB mechanism is ineffective in the presence of traffic of multiple priorities [17]. Efforts are now underway to come up with good priority schemes [11]. However, it seems that the round trip delay will always govern the response time. Therefore, definition (2) will not always be satisfied. Our concern here is how reliable will such a system be if we are dealing with integrated traffic that is transmitted at Gbit/sec rates. Applications, such as voice and video which are delay sensitive should not be delayed by insensitive data. If a certain packet is not transmitted within certain time bounds, this could result in drastic effects on the whole transmission of delay sensitive data.

The most logical approach to this problem is to have a buffering system that can preempt and hold lower priority segments while the higher priority segments are being transmitted. Eventually, the segments that were previously buffered are released in an orderly fashion. A main concern about such a system will be the size of the buffer. Next, we introduce a novel priority scheme that provides the lowest insertion delay for high priority nodes and is still fair to all nodes of the same priority level. Moreover, the buffer size needed is always bounded by a function of the propagation delay.

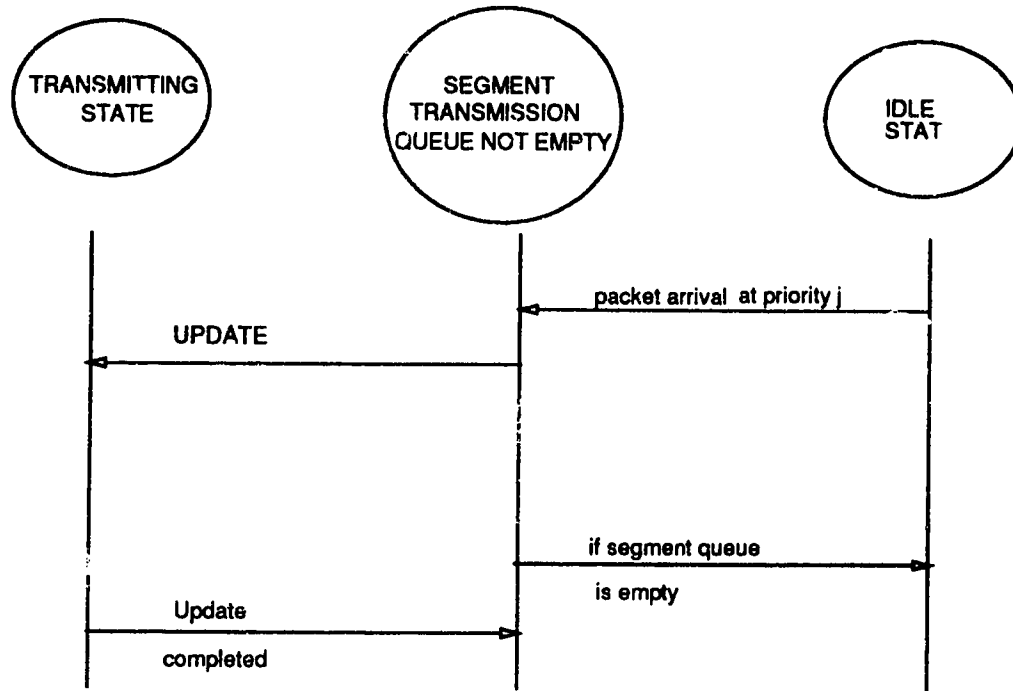
4.2 Priority Scheme for HPDQ

The HPDQ priority scheme, in essence, is the enhanced protocol replicated several times to establish priority levels. To implement this priority scheme, it is necessary that the active switches have buffering capabilities. The operation of this scheme is simple to conceptualize and the finite state machines of nodes operating on the reverse and forward busses are provided in Figure 4.1 and 4.2 respectively. Every node informs the upstream stations that it wants to transmit at a certain priority level. If a node is notified through the reverse bus of the presence of higher priority traffic, it preempts its own transmission on the forward bus until the higher priority transmissions are completed. Thereafter it resumes its transmission. However, if it has higher priority segments queued up for transmission and the upstream stations has already transmitted on the forward bus, it delays the lower priority ones by buffering them and overwriting the slots with its own segments. At the end of its transmission, it will retransmit the segments that it had buffered. Lastly, if its segments are of the same priority level as all other nodes downstream, it enforces the upstream stations to evenly redistribute the bandwidth allocated to nodes of that priority level. Next, we shall elucidate on the implementation of the HPDQ priority scheme. To simplify the

description, we shall impose the following constraint which will be relaxed later. The constraint is that all the buffers from all nodes are always empty. This means that no slots have been buffered at any time. We shall describe the scheme using two priority levels, but it can be easily generalized to multiple levels. Since we have 2 priority levels, we shall assume that α_2 is of higher priority than α_1 . The slot format is shown in Figure 4.3. In addition to the NAND field, each slot will have one Request bit field for each priority level. Request(α_i) represents the request bit for priority level i . The Request(α_i) field is updated on a per segment basis for segments of priority level α_i .

The HPDQ priority scheme herein is described with respect to the enhanced protocol. Nodes will have an AND_ctr, a CD_ctr and a CumReq_ctr for each priority level. The AND_ctr(α_j) counter represents the number of active nodes downstream transmitting at priority level α_j , CD_ctr(α_j) is the CountDown counter for priority level α_j and the CumReq_ctr(α_j) is the number of segments of priority level α_j queued up by downstream stations.

The node updates the NAND and the Request fields based on the values in the Request bit fields and the priority level of the segments queued up at the nodes for transmission. The NAND field is updated differently depending on the value of the priority level of the highest Request field with a bit set. This value, referred to as α is summarized in Table 4.1. The pseudocode for updating the CumReq_ctr, the Request bit field and the NAND field of a slot is given next:



UPDATE = conditions are as indicated in the pseudocode

Figure 4.1: Finite state machine for nodes' operation on the reverse bus.

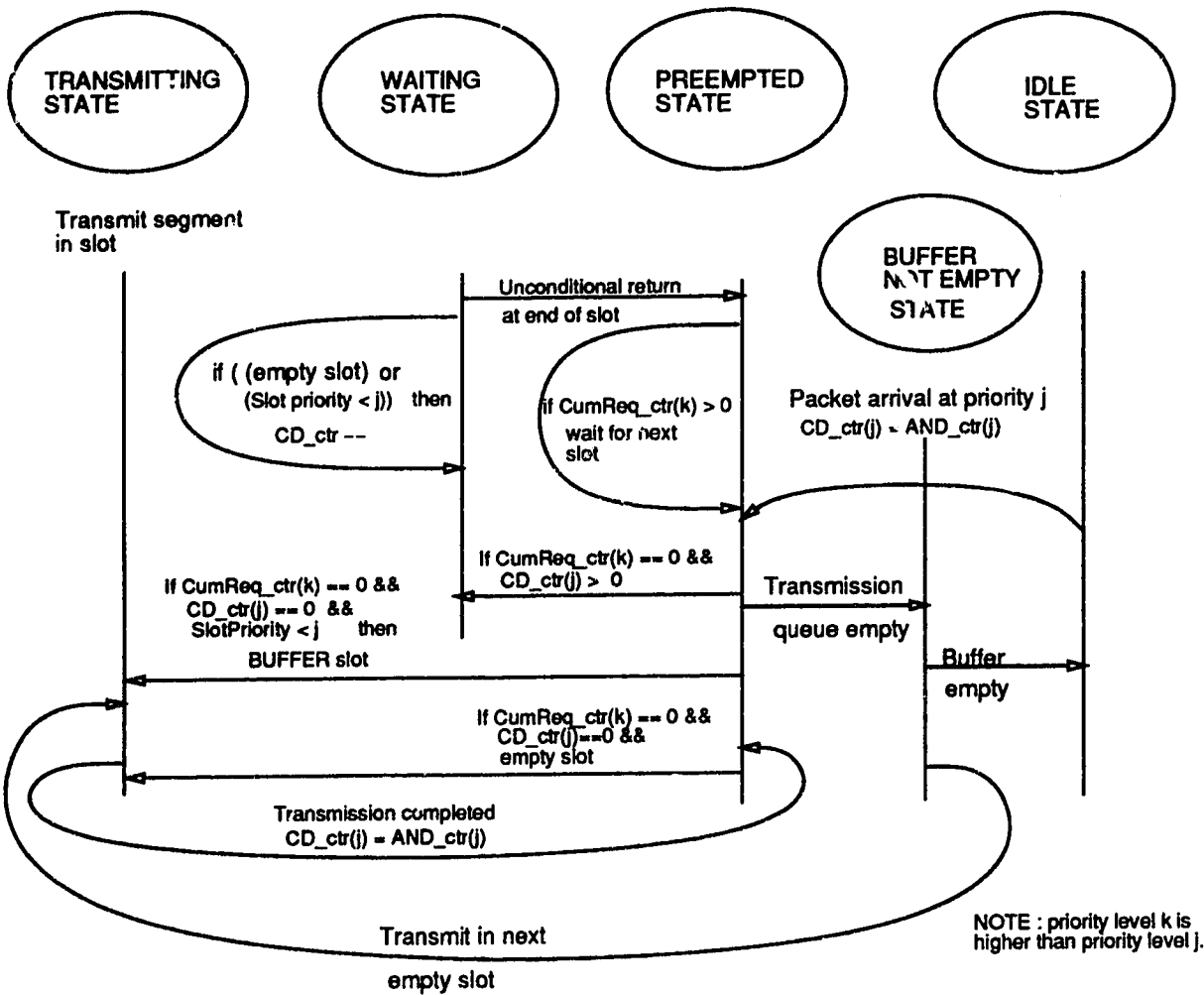
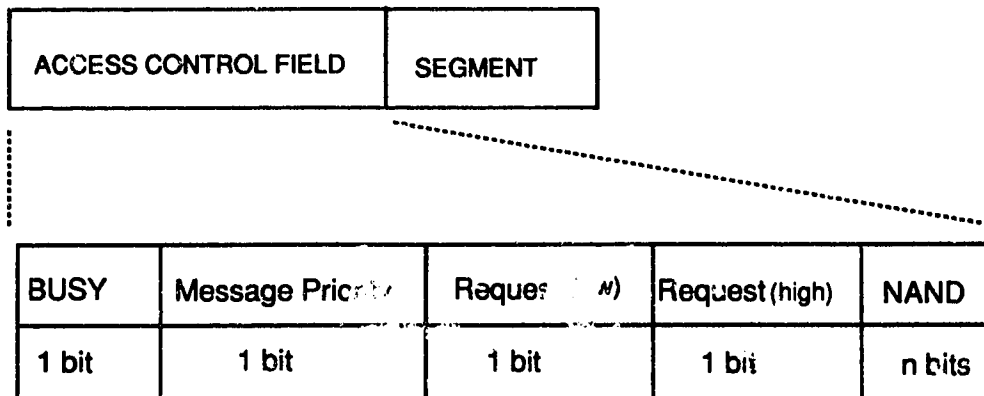


Figure 4.2: Finite state machine for nodes' operation on the forward bus.



$$n = \lceil \log_2 N \rceil$$

Figure 4.3: HPIQ Access Control Field format.

```

UPDATE.REVERSE.BUS()
{
int PriorityLevel, HPD ;
For (PriorityLevel= 1 to 2 )
    if IS_IT_SET(SLOTLR->Request[PriorityLevel])
        /* if the request bit is set */
        CumReq_ctr[PriorityLevel]++
If (SegmentRequestQ[SegmentPriority] > 0 )
    /* request priority is not empty*/
    if (! IS_IT_SET(SLOTLR->Request[SegmentPriority]))
        /*request bit not set*/
        SET(SLOTLR->Request[SegmentPriority])
        SegmentRequestQ[SegmentPriority] - -;
If (SegmentTransmissionQ[SegmentPriority] > )
    HPD = HighestPriorityDownstream();
    If (SegmentPriority == HPD) /* Case 1*/
        NAND++;
    If (SegmentPriority < HPD) /* Case 2 */
        NO CHANGE
    If (SegmentPriority > HPD) /* Case 3 */
        NAND = 1;
}

```

Request(α_1)	Request(α_2)	α
0	0	0
0	1	2
1	0	1
1	1	2

Table 4.1: Summary of α

α	AND_ctr(α_1)	AND_ctr(α_2)
0	0	0
1	NAND	0
2	Unchanged	NAND

Table 4.2: Summary of AND_ctrs update

It is to be noted that when case 2 occurs, the node is prevented from doing any further transmission on the forward bus. In case 1, downstream nodes are transmitting at the same priority level. In case 2 downstream nodes are transmitting segments of high priority and lastly, in case 3, downstream nodes are transmitting segments of lower priority. If a node wants to set a Request bit of matching priority as the SegmentPriority but that bit has been set already then it tries to set that same bit in the next slot. This is repeated recursively until the operation is completed. Table 4.3 summarizes the update of the AND_ctrs before the NAND field is modified.

After every segment transmission or at the beginning of a message arrival to an empty queue at the node, the $\text{AND_ctr}(\text{SegmentPriority})$ is copied into $\text{CD_ctr}(\text{SegmentPriority})$. The $\text{CD_ctr}(\text{SegmentPriority})$ is decremented by one for every empty slot or a lower priority slot that passes by on the reverse bus. Since α_2 is of higher priority than α_1 , α_2 priority level traffic can preempt α_1 priority level traffic. When the $\text{CD_ctr}(\text{SegmentPriority})$ reaches zero, the node schedules a transmission in the next empty slot. Also, if the next slot is of lower priority, the node will transmit its segment by overwriting the segment in that slot, after it has been buffered. This will be explained in more details in the next section.

The CumReq_ctr for each priority level operates in exactly the same way as the one in the unipriority system. $\text{CumReq_ctr}(\alpha_i)$ is incremented by one for each $\text{Request}(\alpha_i)$ bit field which is set. The $\text{CumReq_ctr}(\alpha_i)$ is decremented by one for every slot of priority less than α_i passing by on the forward bus.

4.3 The Buffering System

One of the objectives of a priority scheme is that high priority segments should encounter considerably smaller delay than the lower priority ones [17]. In addition to this, the throughput of higher priority traffic should

be independent of lower priority ones. These are the objectives behind the employment of the buffering system. Now, we shall explain how the buffering system works.

If station i wants to send segments at priority level α_j where $\alpha_j > \alpha_k$, with α_k being the priority level of the slots on the forward bus, then in order to minimize the insertion delay, station i must buffer lower priority segments and then overwrite them with its own segments. At the same time, it will notify the upstream stations of its priority level by updating the NAND field and Request field associated with α_j as explained in the previous section.

Active stations operating at priority level less than α_j will stop transmission after being informed that there are higher priority stations active downstream. In addition to stopping transmission, they also buffer any incoming slots on the forward bus of a priority lower than their own, which effectively implements a buffer insertion scheme. A snapshot of the system after the most upstream station gets the high priority request will reveal that all lower priority stations have shut off their own transmissions and might have buffered some segments, too. Under heavy load traffic conditions, the high priority stations are the only ones operating. The bandwidth is equally divided among them and they are the sole consumers.

After the high priority node has transmitted all its segments at priority level α_j , it stops updating the NAND field on the reverse bus. Thereafter, it transmits the buffered segments. The incoming empty slots or slots of lower

priority are interchanged with previously buffered slots. The above procedure is recursively repeated to release the buffered slots of the upstream nodes.

To remove any ambiguity about this new priority scheme, we consider an example with two nodes separated by 20 slots. Let us assume that station 1 sends low priority traffic and station 2 sends high priority traffic. At t_0 , station 1 is active and has flooded the system with its segments and requests, Figure 4.4(a). Station 2 becomes active at $t_0 + 30$. Since station 2 is a high priority station, it buffers the segments from station 1 and overwrites them with its own segments. meanwhile, it will transmit its high priority requests on the reverse bus. As station 1 is unaware of events downstream, it will keep on transmitting its segments for a while. Figure 4.4(b) shows a snap shot of the network at $t_0 + 50$. Just after this instant, station 1 becomes aware of the transmission of higher priority traffic by the downstream station, and it refrains from any further transmission. In the mean time, station 2 has buffered 20 slots and will buffer 20 more in the next 20 slot times. At $t_0 + 70$, the network configuration looks like that in Figure 4.4(c). In the worst case, the downstream node will need a buffer of size $\lceil 2D \rceil$, where D is the propagation delay between those two nodes measured in slots.

Next, let us consider what happens when the high priority station completes its transmission. After doing so, the high priority station has the responsibility of releasing the buffered slots.

To understand how the buffer is emptied, let us assume that station 2

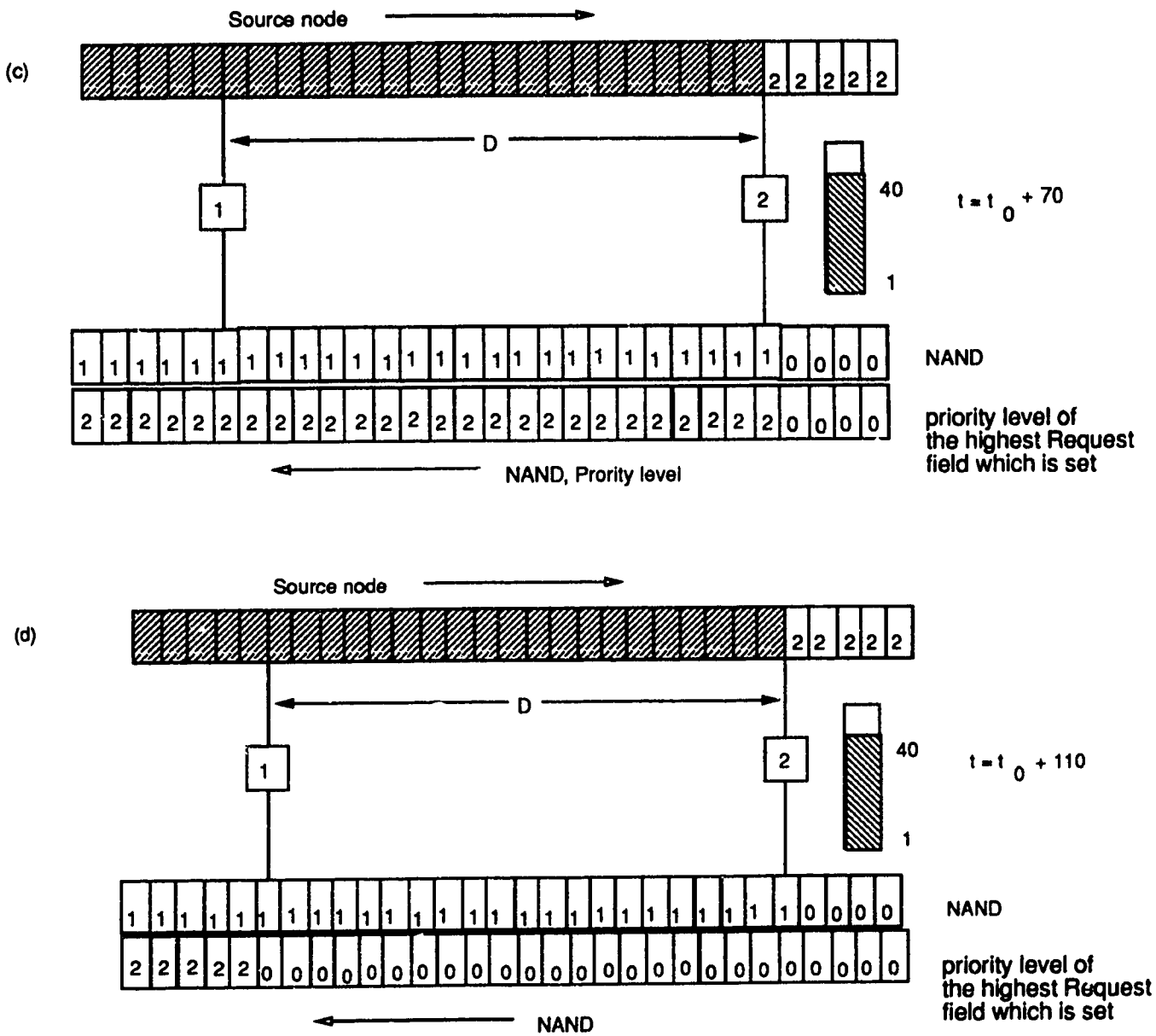


Figure 4.4. : High priority node in operation.

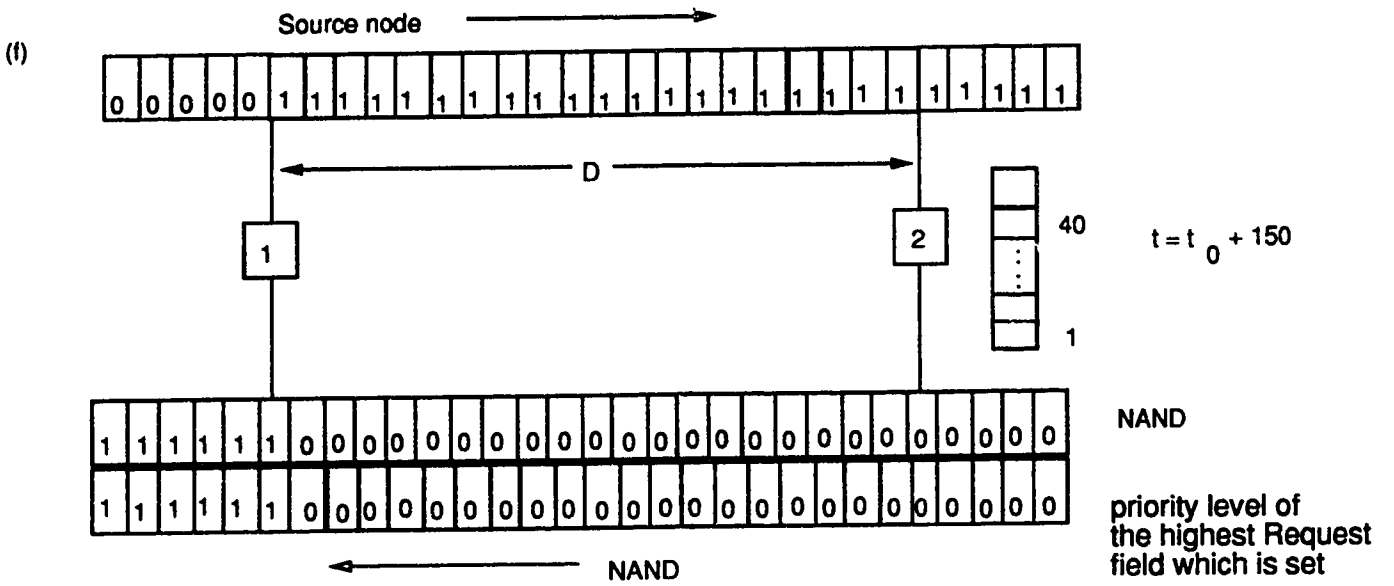
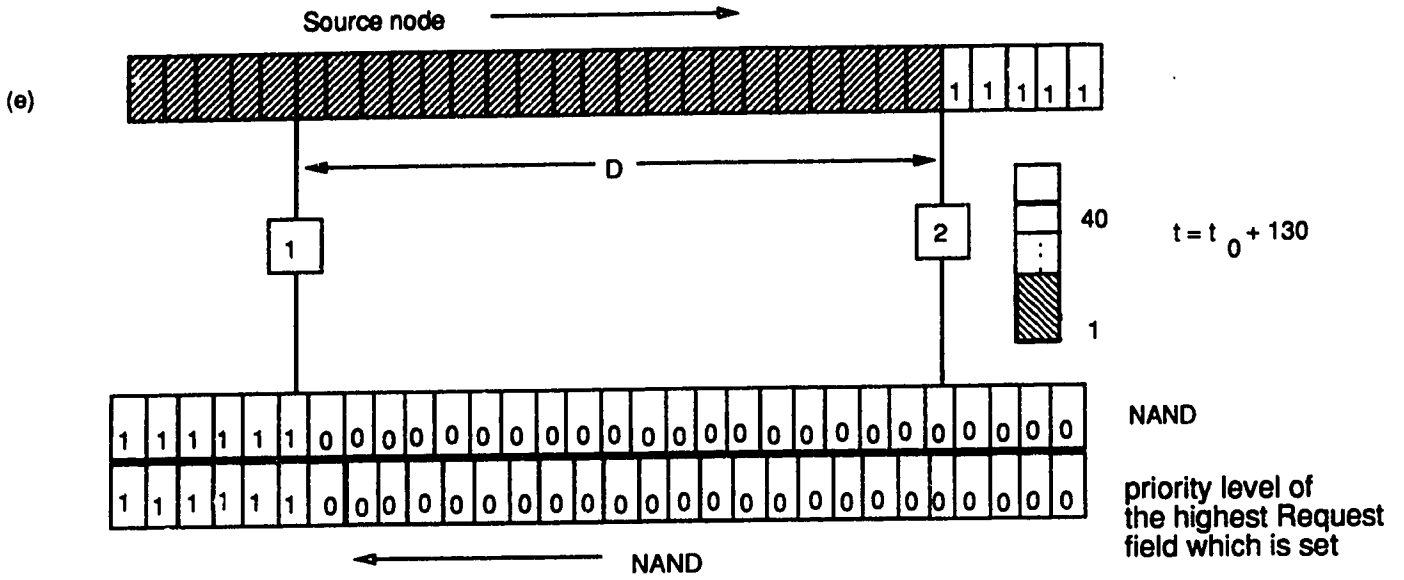


Figure 4.4: High priority node leaving the system

from the previous example stops sending requests on the reverse bus at t_0+90 . Figure 4.4(d) shows the state of the system at $t_0 + 110$. As this information propagates upstream, the slots propagating downstream and dedicated to the α_2 priority class traffic are used to retransmit the buffered slots, shown in Figure 4.4(e). Note that when these free slots reach node 2, their priority is set to that of the buffered slots. When the $\text{CumReq_ctr}(\alpha_2)$ of node 1 is zero, node 1 takes it for granted that node 2 will not require any more empty slots. Hereafter, node 1 will consume all passing slots. Figure 4.4(f) shows the network configuration at $t_0 + 150$. In this simple example, node 2 would have emptied its buffer by the time the clear signal (last updated NAND field from node 2) propagates to the upstream node followed by the node resuming transmission immediately and the first slot reaching node 2.

4.4 Summary

In this chapter we have proposed a new priority scheme for the basic HPDQ protocol. This scheme is novel in the sense that it requires active switches with buffering capabilities. This one is fair to all nodes of the same priority. Additionally, it provides the minimum insertion delay to higher priority traffic. It is also sensitive to neither the internodal distances nor the activity patterns of the nodes. In order to provide the minimum insertion

delay, higher priority nodes are allowed to buffer lower priority segments. Then, the former nodes overwrites the lower priority segments by their own, while buffering the lower priority ones. The segments buffered are deterred for later transmission after the higher priority nodes have completed their own.

Chapter 5

Performance of the HPDQ priority scheme

In chapter 5, a number of simulation results are presented. The goal of these simulations is to demonstrate that the HPDQ protocol with its priority scheme attains a reasonable performance and maintains fairness among nodes when compared to the DQDB protocol. The simulator presented in chapter 3 was modified to accommodate priority. Although the simulator was built for multiple levels of priority, we shall herein study the two priority levels case.

5.1 Nodes with different starting time and positions

In this section we investigate the behavior of the network when a low priority heavy user node enters the system after the system has stabilized. In order to show that the throughput distribution is independent of the position and starting time of a heavy load low priority node, and that the convergence time of bandwidth redistribution is very small and minimal, a simulation study was conducted. The network under study consists of 5 nodes. Two scenarios were considered. In both scenarios, all nodes generate statistical traffic consisting of 24 slot packets including an overhead of 0.17% per slot. All nodes operate at the low priority level except for node 3 which generate low priority traffic in the first case, and high priority traffic in the second. The first message from the high priority node is transmitted after the bus has been filled with lower priority data. The results were similar for the various speeds and are summarized in Table 5.1. From the above simulation results shown in Table 5.1, for the case when node 3 is in high priority mode one can conclude that once the system reaches steady state, the high priority node obtains all the bandwidth it needs, as expected. In this particular case, when the high priority station node 3 becomes active, it shuts off all other nodes completely. The same experiment was repeated for the nodes starting in different orders and at different times. As expected, the steady state

Measurement	Priority of node 3	Utilization				
		0	1	2	3	4
Average Utilization	L	0.2	0.2	0.2	0.2	0.2
	H	0.0	0.0	0.0	1.0	0.0
Average Access time	L	120	120	120	120	120
	H	0	0	0	24	0

Table 5.1: Average utilization and average access delay for various speeds.

behavior was the same in all cases and similar to that in Table 5.1. It can therefore be concluded that the steady state behavior of this priority scheme is insensitive to the nodes positions and actual transmission times. The table also reports the average access time which is defined as the time required to transmit the message once it reaches the head of the queue until its transmission is complete. It is clear from the table that node 3 has a lower access delay when it operates at a higher priority level. This means that the buffering system is helpful in decreasing the access delay of the high priority nodes.

5.2 Nodes of the same priority level

Another important issue which must be considered is how the bandwidth is distributed among nodes of the same priority level. To illustrate that fairness is still achieved amongst high priority nodes, two different scenarios were experimented with. In the first one, taken from [17], there are

8 stations with the two extreme ones generating high priority traffic. The second scenario considers a group of high priority nodes that falls between lower priority ones. Just like in the previous case, the nodes generate statistical traffic consisting of 24 slot packets.

In our first scenario, the high priority nodes uniformly generate traffic that occupies 50% of the bandwidth. Gradually, the lower priority nodes uniformly increase the load they offer to the system. Tables 5.2 and 5.3 provide the results of simulating scenario 1. They show the utilization and average access delay figures, respectively. Note that nodes of the same priority class achieved the same utilization level. This indicates that the protocol is fair to all nodes of the same priority class. Besides, if there is extra bandwidth available after the high priority stations have satisfied their needs, the additional bandwidth is equally shared by all lower priority nodes. Similarly, the mean access delay of higher priority stations are considerably lower than that of the lower priority ones. Within the same priority class, the average access delay of the high priority node remains more or less unchanged. It can be inferred from this behavior that under this priority scheme, high priority demands are almost insensitive to lower priority ones. Tables 5.4 and 5.5 show the average access delay figures of nodes 6 and 7 implementing the DQDB protocol with priority, with and without the Bandwidth Balancing Mechanism. It is to be noted that when the BWB mechanism is used, the global information priority scheme is assumed. As can be observed, the ac-

Bus Load	Utilization							
	0 H	1 H	2 L	3 L	4 L	5 L	6 H	7 H
50	0.125	0.125	0.000	0.000	0.000	0.000	0.125	0.125
70	0.125	0.125	0.050	0.050	0.050	0.050	0.125	0.125
90	0.125	0.125	0.100	0.100	0.100	0.100	0.125	0.125
94	0.125	0.125	0.110	0.110	0.110	0.110	0.125	0.125
97	0.125	0.125	0.118	0.117	0.118	0.117	0.125	0.125
100	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125
107	0.125	0.125	0.125	0.125	0.125	0.125	0.125	0.125

Table 5.2: Utilization for scenario 1.

cess delay under HPDQ is considerably better than that under DQDB with global information priority scheme, especially if the BWB is not used.

In the second scenario, the system consists of five nodes offering the same load to the system. Out of the five nodes only nodes 1, 2, and 3 generate high priority traffic. In order to demonstrate that the low priority nodes do not affect the access time of the high priority nodes, a similar simulation model was constructed in which the low priority nodes offer no traffic at all. Performance results are shown in Table 5.6 and 5.7.

Table 5.6 shows that the high priority nodes equally share the bandwidth. The remaining bandwidth, if any, is distributed evenly among low priority nodes. In this respect, the amount of bandwidth received by the high priority nodes does not depend on the presence of low priority ones. Table 5.7

Bus Load	Average access delay(slots)							
	0 H	1 H	2 L	3 L	4 L	5 L	6 H	7 H
50	36	36	0	0	0	0	37	37
70	36	36	85	85	83	81	37	37
90	36	36	141	142	140	135	37	36
94	36	36	162	162	161	157	38	36
97	36	36	179	179	178	174	37	36
100	36	36	200	197	194	192	37	35
107	36	36	200	198	196	192	38	36

Table 5.3: Average access delay for HPDQ under scenario 1.

Total Load on Bus	Average Access delay	
	Node 0	Node 7
50 %	30	32
70 %	42	47
90 %	77	87
94 %	94	107
97 %	102	121
100 %	123	145
107 %	141	169

Table 5.4: DQDB with priority and without Bandwidth Balancing Mechanism.

Total Load on Bus	Average Access delay	
	Node 6	Node 7
50 %	38	38
70 %	45	47
90 %	61	66
94 %	65	73
97 %	68	75
100 %	68	77
107 %	68	78

Table 5.5: DQDB protocol with Global priority scheme.

Bus Load	Utilization				
	0 L	1 H	2 H	3 H	4 L
50	0.10	0.10	0.10	0.10	0.10
100	0.20	0.20	0.20	0.20	0.20
150	0.05	0.30	0.30	0.30	0.05
200	0.00	0.33	0.33	0.33	0.00
500	0.00	0.33	0.33	0.33	0.00

Table 5.6: (a) Low priority node active .

Bus Load	Utilization				
	0 L	1 H	2 H	3 H	4 L
30	0.00	0.10	0.10	0.10	0.00
60	0.00	0.20	0.20	0.20	0.00
90	0.00	0.30	0.30	0.30	0.00
120	0.00	0.33	0.33	0.33	0.00
300	0.00	0.33	0.33	0.33	0.00

Table 5.6: (b) Low priority nodes inactive.

Table 5.6: Utilization for HPDQ under scenario 2.

Bus Load	Average Access delay(slots)				
	0 L	1 H	2 H	3 H	4 L
50	40	29	29	29	39
100	127	38	38	38	120
150	545	56	56	57	424
200	272	72	72	72	166
500	0	72	72	72	0

Table 5.7: (a) Low priority node active.

Bus Load	Average Access delay(slots)				
	0	1	2	3	4
	L	H	H	H	L
30	0	29	29	29	0
60	0	38	38	38	0
90	0	56	56	56	0
120	0	72	72	72	0
300	0	72	72	72	0

Table 5.7: (b) Low priority node inactive.

Table 5.7: Transfer time for IIPDQ under scenario 2.

indicates that the access time decreases with increasing priority level. Also, comparing Tables 5.7(a) and 5.7(b), it can be additionally deduced that the average access delay encountered by the high priority nodes is similarly unaffected. As stated earlier, this is an attractive feature of this network.

We can therefore conclude that the protocol properly discriminates against lower priority traffic in favor of the higher priority one. It is also fair to all nodes within the same priority level. Fairness is even achieved between lower priority nodes over the bandwidth left over by the high priority ones. This is not surprising since nodes of similar priority levels know exactly how many stations of the same priority level are active downstream.

5.3 Effect of distances on performance of nodes under heavy load

In this section we investigate how the network performance is affected when the internodal distances increases. In DQDB if a low priority node that is upstream from a high priority one has a headon start over the latter, increasing the distance between these nodes is closely related to an increase in the access time and a difference in the bandwidth achieved by the higher priority node. This is understandable as the requests from a high priority node will take longer to propagate to the upstream low priority station before the latter allows one free slot to go by. Surely, as the internodal propagation delay increases so will the access delay.

Since, the operation of HPDC, is fundamentally different from that of DQDB and its priority mechanism employs a buffering mechanism, the performance results under steady state are unaffected by the internodal distances. To verify this statement, a network of 5 nodes with the internodal distances shown in Table 5.8 and a message length of 20 slots was simulated. The high priority nodes were located at different distances downstream from the headend. The simulation results are summarized in Table 5.9. The high priority node shuts off the low priority ones regardless of the node positions. Clearly, this is the case since the lower priority nodes preempt their transmissions in favor of high priority ones as soon as they receives an indication

Node	1	2	3	4	5
Position on the bus (in slots)	0	20	30	40	50

Table 5.8: Internode distances of network of 5 nodes.

Priority	Utilization					Average Delay				
	Node 1	Node 2	Node 3	Node 4	Node 5	Node 1	Node 2	Node 3	Node 4	Node 5
LLLLL	0.2	0.2	0.2	0.2	0.2	100	100	100	100	100
HLLLL	1.0	0.0	0.0	0.0	0.0	20	0.0	0.0	0.0	0.0
LHLLL	0.0	1.0	0.0	0.0	0.0	0.0	20	0.0	0.0	0.0
LLHLL	0.0	0.0	1.0	0.0	0.0	0.0	0.0	20	0.0	0.0
LLLHL	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	20	0.0
LLLLH	0.0	0.0	0.0	0.0	1.0	0.0	0.0	0.0	0.0	20

Table 5.9: Performance results for nodes at different internode distances.

of high priority traffic from the high priority node on the reverse bus.

Table 5.9 also shows that the average delay is independent of the nodes positions and the internodal distances.

In [21], the effect of the different priority node positions were studied. The experiment was carried out on 7 nodes with the $BWB_MOD = 8$. All nodes were operating at overload. A similar scenario was recreated to test the behavior of HPDQ. Tables 5.10 and 5.11 provide simulation results for both DQDB and HPDQ, respectively. The comment field provides observed trends. These tables show that the HPDQ protocol is capable of achieving 100% utilization whereas the DQDB protocol is not. Table 5.10 also shows that for the DQDB protocol higher priority nodes closer to the headend seem to be more favored than those further down [21]. On the other hand, Table

5.11 shows that the high priority nodes of the HPDQ protocol do not behave as those of the DQDB protocol. The high priority nodes implementing the HPDQ protocol, divides the bandwidth fairly among nodes of the same priority level, independent of their locations on the network.

Priorities	1	2	3	4	5	6	7	Total	Comments
LLLLLL	14.03	14.03	14.03	14.03	14.03	14.03	14.03	98.21	H-users at the end have same throughput as L-users.
LLLLLH	14.03	14.03	14.03	14.03	14.03	14.03	<u>14.03</u>	98.21	
LHHHHH	14.03	<u>14.03</u>	<u>14.3</u>	<u>14.03</u>	<u>14.03</u>	<u>14.03</u>	<u>14.03</u>	98.21	
HHHHHH	<u>14.03</u>	<u>14.03</u>	<u>14.03</u>	<u>14.03</u>	<u>14.03</u>	<u>14.03</u>	<u>14.03</u>	98.21	
HLLLLL	<u>88.89</u>	1.82	1.82	1.82	1.82	1.82	1.82	99.81	Leading H-user
HLLLLH	<u>87.00</u>	2.12	2.12	2.12	2.12	2.12	<u>2.1</u>	99.72	H-user at the end elevates throughput of L-users.
HLLLHH	<u>80.33</u>	3.21	3.21	3.21	<u>3.21</u>	<u>3.21</u>	<u>3.21</u>	99.59	
HLHHHH	<u>59.51</u>	6.61	<u>6.61</u>	<u>6.61</u>	<u>6.61</u>	<u>6.61</u>	<u>6.61</u>	99.17	Total throughput decreases.
HHLLLL	<u>47.06</u>	<u>44.06</u>	1.15	1.15	1.15	1.15	1.15	99.87	Leading H-user group
HHLLLH	<u>24.24</u>	<u>24.24</u>	<u>24.24</u>	<u>24.24</u>	0.97	0.97	0.97	99.87	H-users have same throughput
HHHHHL	<u>16.32</u>	<u>16.32</u>	<u>16.32</u>	<u>16.32</u>	<u>16.32</u>	<u>16.32</u>	1.81	99.73	
HHLLHH	<u>46.40</u>	<u>46.40</u>	1.41	1.41	1.41	1.41	<u>1.41</u>	99.85	L-users have higher throughput when additional H-user at the end
HHHHLH	<u>23.90</u>	<u>23.41</u>	<u>23.41</u>	<u>23.41</u>	1.41	1.41	<u>1.41</u>	99.80	
HLLHLL	<u>64.95</u>	1.59	1.59	1.59	<u>26.98</u>	1.59	1.59	99.86	Two H-user groups
HLHHLL	<u>31.61</u>	1.26	<u>21.48</u>	<u>21.48</u>	<u>21.48</u>	1.26	1.26	99.83	Leading H-user have higher throughput
HLLHLH	<u>70.03</u>	2.12	2.12	2.12	<u>19.10</u>	2.12	<u>2.12</u>	99.73	H-user at both ends.
HLHHHL	<u>36.05</u>	2.12	<u>19.10</u>	<u>19.10</u>	<u>19.10</u>	2.12	<u>2.12</u>	99.71	
HLHLHL	<u>45.34</u>	1.82	<u>3.82</u>	1.82	<u>16.35</u>	1.82	<u>1.82</u>	99.80	Throughputs of H-users decrease with increase position
LHLHLH	1.82	<u>45.34</u>	1.82	<u>30.82</u>	1.82	<u>16.35</u>	1.82	99.80	
LLLLLL	4.32	4.32	4.32	4.32	<u>73.52</u>	4.32	4.32	99.44	Related priority configurations (Leading L-users and/or trailing H-users)
LLLLHH	4.32	4.32	<u>73.52</u>	4.32	4.32	<u>4.32</u>	<u>4.32</u>	99.44	
HLLHHH	<u>73.52</u>	4.32	4.32	<u>4.32</u>	<u>4.32</u>	<u>4.32</u>	<u>4.32</u>	99.44	
LLHLHL	3.22	3.22	3.22	<u>54.56</u>	3.22	<u>28.96</u>	3.22	99.44	configuration kernel HLL
HLHLHH	<u>54.56</u>	3.22	<u>28.96</u>	3.22	<u>3.22</u>	<u>3.22</u>	3.22	99.62	configuration kernel HHL

Table 5.10: DQDB: Throughput distribution for different priority configurations (seven users, balancing ratio 8/9).

5.4 SUMMARY

This chapter provides simulation results that illustrate the behavior of the priority scheme of the HPDQ protocol. The simulation results demonstrate that the protocol adequately implements priority and in fact behaves like an ideal slotted preemptive priority scheme. This new priority scheme is insensitive to the time at which high priority transmissions start. It provides the minimum access delay to high priority traffic. Bandwidth is divided fairly and equally among nodes of the same priority level. At light

Priorities	1	2	3	4	5	6	7	Total	Comments
LLLLLL	14.28	14.28	14.28	14.28	14.28	14.28	14.28	100.00	H-users shut off
LLLLLH	0.0	0.0	0.0	0.0	0.0	0.0	<u>100.00</u>	100.00	L-users and consume.
LLLLHH	0.0	<u>16.66</u>	<u>16.66</u>	<u>16.66</u>	<u>16.66</u>	<u>16.66</u>	<u>16.66</u>	100.00	all the bandwidth
LLLLHHH	14.28	14.28	14.28	14.28	14.28	14.28	14.28	100.00	
HLLLLL	<u>100.00</u>	0.0	0.0	0.0	0.0	0.0	0.0	100.00	Leading H-user
LLLLLH	<u>50.00</u>	0.0	0.0	0.0	0.0	0.0	<u>50.00</u>	100.00	H-user at the end does not
LLLLHH	<u>25.00</u>	0.0	0.0	0.0	<u>25.00</u>	<u>25.00</u>	<u>25.00</u>	100.00	change the throughput of L-users.
HLLLLHH	<u>16.66</u>	0.0	<u>16.66</u>	<u>16.66</u>	<u>16.66</u>	<u>16.66</u>	<u>16.66</u>	100.00	
HHLLLL	<u>50.00</u>	<u>50.00</u>	0.0	0.0	0.0	0.0	0.0	100.00	Leading H-user group
HHHLLL	<u>25.00</u>	<u>25.00</u>	<u>25.00</u>	<u>25.00</u>	0.0	0.0	0.0	100.00	Bandwidth is equally
HHHHLL	<u>16.66</u>	<u>16.66</u>	<u>16.66</u>	<u>16.66</u>	<u>16.66</u>	<u>16.66</u>	0.0	100.00	divided among H-users
HHLLHH	<u>33.33</u>	<u>33.33</u>	0.0	0.0	0.0	0.0	<u>33.33</u>	100.00	No change in throughput of H-users
HHHLLH	<u>20.00</u>	<u>20.00</u>	<u>20.00</u>	<u>20.00</u>	0.0	0.0	<u>20.00</u>	100.00	by addition of new ones at the end.
HLLHLL	<u>50.00</u>	0.0	0.0	0.0	<u>50.00</u>	0.0	0.0	100.00	Two H-user groups
HLHHLL	<u>25.00</u>	0.0	<u>25.00</u>	<u>25.00</u>	<u>25.00</u>	0.0	0.0	100.00	Throughput is insensitive
HLLHLH	<u>33.33</u>	0.0	0.0	0.0	<u>33.33</u>	0.0	<u>33.33</u>	100.00	to the starting time.
HLHHLH	<u>20.00</u>	0.0	<u>20.00</u>	<u>20.00</u>	<u>20.00</u>	0.0	<u>20.00</u>	100.00	H-user at both ends
HLLHLH	<u>25.00</u>	0.0	<u>25.00</u>	0.0	<u>25.00</u>	0.0	<u>25.00</u>	100.00	equally divided throughput
LHLHLH	0.0	<u>33.33</u>	0.0	<u>33.33</u>	0.0	<u>33.33</u>	0.0	100.00	Throughputs of H-users decrease
LHLHLH	0.0	<u>33.33</u>	0.0	<u>33.33</u>	0.0	<u>33.33</u>	0.0	100.00	with increasing number of H-users
LLLHLH	0.0	0.0	0.0	0.0	<u>100.00</u>	0.0	0.0	100.00	Related priority configurations
LLHLHH	0.0	0.0	<u>33.33</u>	0.0	0.0	<u>33.33</u>	<u>33.33</u>	100.00	(Leading L-users and/or
HLHHHH	<u>20.00</u>	0.0	0.0	<u>20.00</u>	<u>20.00</u>	<u>20.00</u>	<u>20.00</u>	100.00	trailing H-users)
LLLHLH	0.0	0.0	0.0	<u>50.00</u>	0.0	<u>50.00</u>	0.0	100.00	Throughput fairly and
HLHLHH	<u>20.00</u>	0.0	<u>20.00</u>	0.0	<u>20.00</u>	<u>20.00</u>	<u>20.00</u>	100.00	equally divided

Table 5.11: HPDQ: Throughput distribution for different priority configurations.

load, the bandwidth that is unconsumed by high priority nodes is divided equally among nodes with the next lower priority level. This procedure is recursively repeated until either the bus bandwidth is exhausted or all nodes have satisfied their bandwidth requirements. At overload, the high priority stations will shut off the lower priority ones. Our simulation results also reveal that the new protocol is insensitive to internodal distances. The spectrum of simulation covered in this chapter justifies that the HPDQ protocol with its priority scheme provides an acceptable behavior and performance.

Chapter 6

CONCLUSIONS and DIRECTIONS for FUTURE RESEARCH

In this thesis we have described the problems that typical MAC protocols designed for BISDN suffer from. The Standard IEEE 802.6 MAC protocol (DQDB) was taken as an example of many such protocols. The DQDB protocol seems to distribute bandwidth unfairly in both single and multiple priority traffic systems. The unipriority system is sensitive to the order in which nodes start their transmissions. The multi-priority schemes proposed for DQDB seem to inherit similar problems. The throughput of higher priority nodes might sometimes depend on the location and starting

times of lower priority nodes which is an undesirable behavioral feature. Besides, the priority scheme might exhibit an inverse effect which means that the throughput of a lower priority node drops when it changes to a higher priority level. The IEEE 802.6 committee adopted the Bandwidth Balancing Mechanism (BWB) to alleviate the aforementioned problems, but that did not solve the problems completely. Under the Bandwidth Balancing Mechanism, part of the network bandwidth is sacrificed in order to provide a coherent coordination among nodes which enable nodes to achieve a fairer distribution of bandwidth. A derivative of the BWB called global information priority scheme has been proposed for problems pertaining to multi priority traffic. The results obtained do not overcome all the problems, but the mechanism mentioned above provides a noticeable improvement.

In this thesis we have described a novel protocol that we referred to as the High Performance Distributed Queue. The HPDQ protocol is fair to all nodes and the bandwidth distribution among nodes is insensitive to the time at which a node starts its transmission. The bandwidth is divided equally among nodes of the same priority. The only requirement of the HPDQ protocol is that active switches be used to tap nodes on the bus. The active switch should also have the ability to buffer incoming slots and modify them.

A priority scheme based on HPDQ is also described in this thesis. The proposed priority scheme is fair to nodes of all priority levels. The scheme effectively discriminates between nodes of different priority levels in favor of the

higher priority nodes, while still being fair to nodes of the same priority level. Under all loading conditions, high priority nodes satisfy all their bandwidth requirements without any intervention from the low priority ones. Bandwidth distribution among nodes of the same priority level is proportional to the load of traffic that each node is adding to the bus. Any remaining bandwidth is assigned to lower priority nodes. The same procedure for bandwidth allocation is repeated for nodes of lower priority. Further investigation showed that the high priority nodes are totally independent of lower priority ones. Besides, the priority scheme is also insensitive to the time at which the nodes start transmitting. As a byproduct of the above, the high priority nodes will shut off the lower priority ones, under overload conditions. To minimize insertion time a buffering mechanism was proposed. This buffering mechanism requires that the switches be equipped with a buffer of size $\lceil 2D \rceil$, where D is the end-to-end propagation delay in terms of slots. Simulation results showed that the objectives of the protocol are fully achieved. In order to eliminate the loss of bandwidth incurred by nodes leaving the system, an enhanced version of the basic protocol was proposed. Because of its full use of bandwidth, fairness among nodes, low access delay and effective priority implementation, it can be said that HPDQ envelops the qualities of a good protocol.

Simulation experiments with the HPDQ protocol did not reveal

any symptoms of any kind of problems. However, the bandwidth utilization can be improved by designing an appropriate channel reuse mechanism. This should not be hard to implement since the hardware configuration of the HPDQ network includes all the necessary components for slot reuse. What remains to be done is designing the reuse protocol. We can also improve the performance of the protocol by decreasing the overhead. Slot overhead can be reduced if we limit our NAND field to serve less than the maximum number of stations on the network. In case the number of active stations exceed the new limit, the system should have a mechanism to assign bandwidth. Another possible improvement could be to investigate the possibility of attaining similar performance results with simpler hardware.

Bibliography

- [1] H. Armbruster and G. Arndt. "Broadband Communication and its Realization with Broadband ISDN". *IEEE Communication Magazine*. Vol 25. No 11. November, pages 8-19, 1987.
- [2] J. Filipiak. "Access Protection for Fairness of Distributed Queue Dual Bus Metropolitan Area Network". *ICC*, pages 635-639, 1989.
- [3] IEEE 802.6 Working Group. "IEEE Standard 802.6: Distributed Queue Dual Bus Subnetwork of a Metropolitan Area Network". *IEEE*, Oct, 1990.
- [4] B. W. Abeyesundara and A. E. Kamal. "High Speed Local Area Networks and their Performance: A survey". *ACM, Computing Survey*, June 1991.
- [5] S. Kano and K. Kitami and M. Kawarasaki. "ISDN Standardization.". *Proc. IEEE, Vol 79, No 2, February*, pages 118-124, 1991.

- [6] L. Kleinrock. "ISDN-The path to Broadband Networks.". *Proc of the IEEE. Vol 79. No 2. February*, pages 112-117, 1991.
- [7] M. Conti and E. Gregori and L. Lenzini. "A Methodological Approach to an Extensive Analysis of DQDB Performance and Fairness.". *IEEE SAC. Vol 9. NO 1. January*, pages 76-85, 1991.
- [8] M. Conti, E. Gregori and L. Lenzini. "A Methodological Approach to an Extensive Analysis of DQDB Performance and Fairness.". *IEEE Journal on Selected Areas in Communication. Vol 9. No 1*, pages 76-86, Jan 1991.
- [9] M. Conti, E. Gregori and L. Lenzini. "Simulation study of Dual Bus MAN Protocols". *3rd IEEE Workshop on Metropolitan Area Networks, San Diego*, pages 375-408, March 1989.
- [10] E. L. Hahne, A. K. Chaudhury and N. F. Maxemchuk. "Improving the fairness of Distributed Queue Dual bus Networks ". *Infocom*, 1990.
- [11] E. L. Hahne and N. F. Maxemchuk. "Bandwidth Balancing with Global Priority Information". *IEEE 802.6 Contribution to Tuscon 1990 meeting*.
- [12] E. L. Hahne and N. F. Maxemchuk. "Fair access of multi-priority traffic to distributed queue dual bus networks.". *Proc. INFOCOM*, pages 889-900, 1991.

- [13] E. L. Hahne and A. K. Choudhury and N. F. Maxemhuck. "Improving The Fairness Of Distributed-Queue-Dual-Bus Networks.". *IEEE*, pages 175-184, 1990.
- [14] James F. Mollenauer. "Standards for Metropolitan Area Network". *IEEE Communication Magazine*, Vol. 26, No 4., pages 15-19, 1988.
- [15] M. Mehdi Nassehi. "CRMA: An Access scheme for High Speed LANs and MANs". *INFOCOM*, pages 1697-1702, 1990.
- [16] M. J. Rider. "Protocols for ATM Access Networks.". *INFOCOM*, pages 112-117, 1988.
- [17] M. Spratt. "The Effectiveness of priority in the IEEE 802.6 protocol.". *Contribution to the IEEE 802.6 working group.*, 1990.
- [18] A. Tanenbaum. "Computer Networks.". *Second edition, Prentice Hall*, 1988.
- [19] C. Bisdikian and A. N. Tantawy. "A Mechanism for Implementing preemptive Priorities in DQDB Subnetworks.". *ICC*, pages 1062-1067, 1991.
- [20] F. Tobagi. "Fast Packet Switch Architectures For Broadband Integrated Services Digital Networks". *IEEE SAC. Vol 78. No 1. January*, pages 133-167, 1990.

- [21] H. R. van As. "Performance Evaluation of Bandwidth Balancing in the DQDB Mac Protocol". *EFOC/LAN 90*, pages 231--239, 1990.
- [22] H. R. van As, J. W. Wong and P. Zafropulo. "Fairness, Priority and Predictability of the DQDB MAC Protocol Under Heavy Load". *Int'l Zurich Seminar on Digital Communications Electronic Circuits and System for communication*, 1990.
- [23] H. R. van As, W. W. Lempenau, P. Zafropulo and E. A. Zurfluh. "CRMA-2: A Gbit/s MAC Protocol for ring and Bus Networks with Immediate Access Capability". *EFOC/LAN*, pages 56--71, 1991.
- [24] H. R. van As, W. W. Lempenau, P. Zafropulo and E. A. Zurfluh. "Performance of CRMA-2: A Reservation-Based Fair Media Access Protocol for Gbits/s LANs and MANs with buffer Insertion". *EFOC/LAN*, pages 1-8, 1992.
- [25] M. Kabatepe and K. S. Vastola. "FDQ: The Fair Distributed Queue MAN". *INFOCOM*, pages 200-209, 1992.
- [26] P. Davids and Th. Welzel. "Performance Analysis of DQDB Based on Simulation". *3rd IEEE Workshop on Metropolitan Area Networks*, pages 431-445, March 1989.
- [27] J .W. Wong. "Throughput of DQDB Networks Under Heavy Load". *EFOC/LAN*, 1989.

- [28] Hans R. Muller, M. Mehdi Nassehi, Johnny W. Wong, Erwin Zurfluh, Werner Bux and Pitro Zafropulo. "DQMA and CRMA: New Access Schemes for Gbit/s LANs and MANs". *ICC*, pages 185–191, 1990.